aws

Guide du développeur

Amazon Elastic Container Service



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

Amazon Elastic Container Service: Guide du développeur

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

Les marques et les présentations commerciales d'Amazon ne peuvent être utilisées en relation avec un produit ou un service extérieur à Amazon, d'une manière susceptible de créer une confusion chez les clients, ou d'une manière qui dénigre ou discrédite Amazon. Toutes les autres marques commerciales qui ne sont pas la propriété d’Amazon appartiennent à leurs propriétaires respectifs, qui peuvent ou non être affiliés ou connectés à Amazon, ou sponsorisés par Amazon.

Table of Contents

Qu'est-ce que Amazon ECS ? 1
Terminologie et composants 1
Caractéristiques 3
Allouer 4
Tarification 4
Services connexes 4
Prise en main 6
Configuration 6
Console de gestion AWS 6
Inscrivez-vous pour Compte AWS 7
Créer un Virtual Private Cloud 7
Création d’un groupe de sécurité 8
Créer les informations d'identification pour vous connecter à votre instance EC2 13
Installer la AWS CLI 13
Prochaines étapes pour l’utilisation d’Amazon ECS 14
Création d'une image de conteneur 15
Conditions préalables 15
Créer une image Docker 17
Transmettre votre image à Amazon Elastic Container Registry 19
Nettoyage 21
Étapes suivantes 21
Création d’une tâche pour les instances gérées Amazon ECS 21
Conditions préalables 21
Étape 1 : créer un cluster 22
Étape 2 : Créer une définition de tâche 23
Étape 3 : Création d'un service 24
Étape 4 : Afficher votre service 25
Étape 5 : nettoyer 25
Apprenez à créer une tâche pour les instances gérées Amazon ECS à l'aide du AWS CLI 26
Conditions préalables 27
Étape 1 : créer un cluster 28
Étape 2 : créer un fournisseur de capacité pour les instances gérées Amazon ECS 29
Étape 3 : définir une stratégie de fournisseur de capacité par défaut pour le cluster 30
Étape 4 : enregistrer une définition de tâche Linux 30

Étape 5 : répertorier les définitions de tâche 32
Étape 6 : créer un service 32
Étape 7 : répertorier les services 33
Étape 8 : décrire le service en cours d'exécution 33
Étape 9 : tester 35
Étape 10 : nettoyer 37
Création d'une tâche Linux pour Fargate 38
Conditions préalables 38
Étape 1 : créer le cluster 39
Étape 2 : Créer une définition de tâche 40
Étape 3 : Créer le service 41
Étape 4 : Afficher votre service 42
Étape 5 : nettoyer 42
Création d'une tâche Windows pour Fargate 43
Conditions préalables 43
Étape 1 : créer un cluster 44
Étape 2 : Enregistrer une définition de tâche Windows 44
Étape 3 : Créer un service avec votre définition de tâche 46
Étape 4 : Afficher votre service 46
Étape 5 : Nettoyer 47
Création d'une tâche Windows pour EC2 47
Conditions préalables 48
Étape 1 : créer un cluster 49
Étape 2 : Enregistrer une définition de tâche 50
Étape 3 : Créer un service 52
Étape 4 : Afficher votre service 52
Étape 5 : Nettoyer 53
À l'aide du AWS CDK 54
Étape 1 : configurez votre AWS CDK project 55
Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate 57
Étape 3 : Tester le service web 65
Étape 4 : nettoyer 65
Étapes suivantes 66
Création de ressources à l'aide du AWS Interface de ligne de commande Copilot 67
Installation de la AWS Interface de ligne de commande Copilot 68

Déploiement d'un exemple d'application Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot 77
Amazon ECS avec CloudFormation 80
Création de ressources Amazon ECS à l'aide de la CloudFormation console 81
Conditions préalables 81
Étape 1 : créer un modèle 81
Étape 2 : créer une pile pour les ressources Amazon ECS 93
Étape 3 : vérifier 94
Étape 4 : Nettoyer les ressources 95
Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour CloudFormation 95
Conditions préalables 95
Étape 1 : créer une pile 96
Étape 2 : vérifier la création de ressource Amazon ECS 108
Étape 3 : Nettoyer 110
Exemples de CloudFormation modèles 110
Définitions de tâche 111
Fournisseurs de capacité 123
Clusters 125
Services 138
Rôles IAM pour Amazon ECS 212
Bonnes pratiques 216
Architecture de votre solution pour Amazon ECS 221
Capacity 222
Points de terminaison de service 223
Mise en réseau 223
Accès aux fonctionnalités 224
Rôles IAM 225
Logging 226
Pratiques exemplaires relatives aux images de conteneur 226
Types de lancements et fournisseurs de capacités 226
Bonnes pratiques 227
Mutabilité des services 228
Utilisation des points de terminaison Dual-Stack 230
Utilisation des points de terminaison Dual-Stack (AWS CLI) 231
Utilisation de points de terminaison à double pile (SDK)AWS 231

Utilisation de points de terminaison Dual-Stack avec l'API REST 232
Applications dans des sous-réseaux partagés, des zones locales et des Zones Wavelength 233
Sous-réseaux partagés 233
Local Zones 235
Zones Wavelength 235
Amazon Elastic Container Service sur AWS Outposts 236
Considérations 236
Conditions préalables 237
Vue d'ensemble de la création de clusters sur AWS Outposts 237
Pratiques exemplaires en matière d’images de conteneur 239
Bonnes pratiques concernant le réseau 242
Connexion d’applications à Internet 242
Pratiques exemplaires en matière de réception de connexions entrantes vers Amazon ECS 246
Meilleures pratiques pour se connecter aux AWS services 252
Pratiques exemplaires de connexion des services 255
exemplaires en matière de services de mise en réseau entre les AWS et les VPC 260
AWS services de dépannage réseau 261
Optimisation du temps de lancement des tâches 262
Exploitation à grande échelle 264
Quotas de service Amazon ECS et seuils de limitation d’API 264
Gestion des problèmes de limitation 270
Pratiques exemplaires en matière d’autoscaling et de gestion des capacités 273
Détermination de la taille d’une tâche 274
Optimisation de l’autoscaling du service 275
Capacité et disponibilité 282
Capacité de cluster 286
Choix de la taille des tâches Fargate 287
Accélération de l’autoscaling de cluster 288
Accès aux fonctionnalités avec les paramètres du compte 290
Amazon Resource Names (ARN) et ID 291
Calendrier relatif au format ARN et de l'ID de ressource 293
Container Insights 294
AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140) 296
Autorisation de balisage 297
Chronologie des autorisations de balisage 298

AWS Fargate temps d'attente pour le retrait d'une tâche 299
AWS Fargate retraits de tâches à l'aide des fenêtres d'événements EC2 300
Augmentation des interfaces réseau des instances de conteneur Linux 301
Surveillance de l'exécution (GuardDuty intégration Amazon) 301
VPC IPv6 à double pile 302
Mode de pilote du journal par défaut 303
Affichage de vos paramètres de compte à l'aide de la console 304
Modification des paramètres de compte 304
Restauration des paramètres par défaut du compte 305
Gérer les paramètres du compte à l'aide du AWS CLI 306
Rôles IAM pour Amazon ECS 308
Instances gérées Amazon ECS 313
Prise en main 313
Fournisseurs de capacité 314
Sélection et optimisation des instances 314
Définitions de tâche 315
Système d’exploitation et architecture de l’UC 316
Fonctionnalités principales 316
Rôles IAM 316
Conformité et sécurité 317
Mise en réseau 318
Stockage d’instances 318
Stockage local 319
Équilibrage de charge des services 322
Surveillance et observabilité 322
Tarification et optimisation des coûts 323
Quotas de service 323
Considérations concernant la migration 323
Limites et considérations 324
Contrôles de l'organisation 324
Exemple de SCP pour bloquer toutes les AMI à l'exception des AMI Amazon ECS Managed Instances 325
Comptes AMI Amazon ECS Managed Instances 326
Types d’instances gérées Amazon ECS 327
Méthodes de sélection d’instances 328
Sélection de type d’instance spécifique 328

Attribute-based sélection du type d'instance 328
Options de facturation et d’achat 329
Types d'instances pris en charge par Amazon ECS Managed Instances 330
Pratiques exemplaires en matière de sélection d’instances pour les instances gérées Amazon ECS 331
Comportement d'extraction des instances gérées Amazon ECS 333
Résolution des images de conteneurs 333
Application de correctifs dans les instances gérées Amazon ECS 334
Cycle de vie d’une instance 334
Fenêtres d’événements et planification de la maintenance 334
Séquence de maintenance 335
Considérations de sécurité pour les instances gérées Amazon ECS 335
Modèle de sécurité 336
Comprendre les instances gérées 336
Fonctions de sécurité 337
Support en matière de conformité et de réglementation 338
Instances gérées Amazon ECS FIPS-140 Considérations 339
Désactiver FIPS sur les instances gérées Amazon ECS 339
Considérations sur la sécurité 340
Bonnes pratiques de sécurité 341
Activation du contrôle du chiffrement VPC pour les instances gérées Amazon ECS 342
Conditions préalables 342
Identifier les types d'instances compatibles 343
Création d'un cluster avec prise en charge du chiffrement VPC 343
Considérations 344
Résolution des problèmes 344
Optimisation de l’infrastructure 344
Détection des instances inactives 345
Détection des instances sous-utilisées 345
ScaleInAfter 346
Réparation automatique du GPU 347
Comment ça marche 347
Surveillance de l'état du GPU 348
Codes d'erreur XID surveillés 348
Désactivation de la réparation automatique 349
Migration de Fargate vers les instances gérées Amazon ECS 351

Considérations concernant la migration 351
Conditions préalables 352
Étape 1 : mettre à jour le cluster pour utiliser les instances gérées Amazon ECS 353
Étape 2 : mettre à jour la définition de tâche pour bénéficier de la fonctionnalité d'instances gérées Amazon ECS 353
Étape 3 : mettre à jour le service pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS 353
Étape 4 : procéder à la migration des tâches autonomes 353
Migration d'EC2 vers les instances gérées Amazon ECS 354
Considérations concernant la migration 354
Conditions préalables 355
Étape 1 : mettre à jour le cluster pour utiliser les instances gérées Amazon ECS 355
Étape 2 : mettre à jour la définition de tâche pour bénéficier de la fonctionnalité d'instances gérées Amazon ECS 355
Étape 3 : mettre à jour le service pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS 356
Étape 4 : procéder à la migration des tâches autonomes 356
Démons gérés par Amazon ECS 357
Criticité des démons 357
Comment fonctionnent les démons gérés 357
Principaux avantages 358
Prise en main 359
Définitions de tâches de démon 359
Création d'une définition de tâche de démon 360
Paramètres de définition des tâches du démon 361
Mode réseau 362
Capacités privilégiées 362
Partage d'espaces de noms PID et IPC 363
Volumes 365
Paramètres de conteneur pris en charge 366
Création et gestion de démons 367
Conditions préalables 367
Création d'un démon 367
Vérification du déploiement du démon 369
Mettre à jour un démon 370
Supprimer un démon 371

Déploiements de démons 372
Comment fonctionnent les déploiements de démons 372
Paramètres de configuration du déploiement 373
Surveillance des déploiements 373
Scénarios de déploiement de démons 374
Événements Daemon 377
Événements d'action du service Daemon 378
Événements de changement d'état de déploiement du démon 380
Création de règles pour les événements du démon 382
Métriques des démons 383
Métriques d'utilisation des démons 383
Affichage des métriques des démons 384
Réparation automatique Daemon 386
AWS Fargate 388
Procédures 388
Fournisseurs de capacité 389
Définitions de tâche 389
Versions de plateforme 389
Équilibrage de charge des services 390
Métriques d'utilisation 390
Considérations relatives à la sécurité concernant l'utilisation de Fargate 391
Pratiques exemplaires en matière de sécurité Fargate 391
Utilisation AWS KMS pour crypter le stockage éphémère pour Fargate 391
Fonctionnalité SYS_PTRACE pour le suivi des appels système du noyau avec Fargate 392
Utilisez Amazon GuardDuty avec Fargate Runtime Monitoring 392
Considérations relatives à la sécurité Fargate 393
Versions de plateforme Fargate 394
Migration vers la version de plateforme Linux 1.4.0 396
Journal des modifications de la version de plateforme Linux 397
Obsolescence de la version de plateforme Linux 399
Journal des modifications de la version de plateforme Windows 402
Considérations sur les conteneurs Windows sur Fargate pour Amazon ECS 403
Comportement d'extraction d'image de conteneur Linux sur Fargate 404
Comportement d'extraction d'image de conteneur relatif aux conteneurs Windows sur Fargate 406
Stockage éphémère de la tâche Fargate 406

Versions de la plateforme de conteneurs Linux Fargate 407
Clés gérées par le client pour le stockage éphémère AWS Fargate 408
Mise hors service et maintenance des tâches 421
Aperçu des avis de mise hors service de tâches 422
État de la ressource affectée dans un avis de retrait de tâche 426
Puis-je refuser la mise hors service d'une tâche ? 426
Puis-je recevoir des notifications de suppression de tâches par le biais d'autres AWS des services ? 427
Puis-je modifier la mise hors service d'une tâche une fois qu'elle a été planifiée ? 427
Comment Amazon ECS gère-t-il les tâches faisant partie d'un service ? 428
Amazon ECS peut-il gérer automatiquement des tâches autonomes ? 428
Résolution des problèmes de disponibilité du service pendant la mise hors service d'une tâche 428
Préparez-vous pour AWS Retrait des tâches Fargate sur Amazon ECS 429
AWS Régions de Fargate 435
Conteneurs Linux sur AWS Fargate 435
Conteneurs Windows allumés AWS Fargate 437
EC2 440
Comportement d'extraction d'image de conteneur pour EC2 et les instances externes 441
Amazon ECS Anywhere 443
Mode Amazon ECS Express 444
Avantages des services Amazon ECS Express Mode 444
Tarification 445
Disponibilité 445
Création d'un service en mode Express 446
Conditions préalables 446
Procédures 446
Personnalisation 447
Créez votre premier service Express Mode dans la console 447
Procédure 448
Étapes suivantes 449
Personnalisation 449
Créez votre premier service en mode Express à l'aide du AWS CLI 451
Conditions préalables 451
Étape 1 : Création de rôles IAM 452
Étape 2 : Créez votre première application de service en mode Express 453

Étape 3 : Création d'une application de service en mode Express avec des paramètres personnalisés ... 447
Étape 4 : Surveillez votre déploiement ... 456
Étape 5 : Accédez à votre application ... 456
Ressources créées par les services Express Mode ... 457
Paramètres par défaut du cluster ... 458
Définition des tâches par défaut ... 458
Définition des conteneurs par défaut ... 459
Paramètres de journalisation par défaut ... 459
Paramètres de service par défaut ... 460
Paramètres de configuration réseau par défaut ... 462
Valeurs par défaut du rôle IAM ... 463
Paramètres par défaut de l'application Auto Scaling ... 464
Valeurs par défaut de la cible de dimensionnement des applications ... 464
Paramètres par défaut de la politique de dimensionnement des applications ... 465
Valeurs par défaut de l'équilibreur de charge des applications ... 466
Paramètres par défaut du groupe cible ... 467
Partage des ressources et optimisation des coûts ... 467
Afficher les détails d'un service en mode Express ... 468
Mettre à jour un service Amazon ECS Express Mode ... 469
Comportement de déploiement ... 469
Procédure ... 470
Supprimer les services du mode Express ... 473
Différence entre la suppression d'un service en mode Express et d'un service ... 474
Procédure ... 474
Meilleures pratiques pour les services Amazon ECS Express Mode ... 475
Bonnes pratiques de sécurité ... 475
Optimisation des performances et du calcul ... 477
Meilleures pratiques opérationnelles ... 479
Résolution des problèmes ... 480
Problèmes de déploiement ... 481
Erreurs de définition de tâches personnalisées ... 482
Problèmes de connectivité ... 483
Problèmes de performance ... 484
Outils de surveillance et de débogage ... 485
Mise à jour des ressources en dehors du mode Express ... 486

Comprendre le modèle de responsabilité partagée 486
Dois-je faire quelque chose avant de pouvoir profiter des fonctionnalités d'Amazon ECS dans mon service Express Mode ? 486
Exemples de mises à jour en dehors du mode Express 487
Passerelle Amazon ECS 490
Comment fonctionne Amazon ECS Gateway 490
Avantages d'Amazon ECS Gateway 491
Concepts clés 492
Infrastructure gérée par Amazon ECS Gateway 492
Tarification 493
Disponibilité 494
Limites du service Gateway 494
Considérations 494
Création et gestion d'une passerelle 495
Conditions préalables 496
Créer une passerelle 498
Associer une passerelle à des clusters Amazon ECS 500
Mettre à jour une passerelle 502
Dissocier une passerelle d'un cluster 510
Supprimer une passerelle 512
Connectez les services à une passerelle 513
Conditions préalables 513
Connecter un service à une passerelle 514
Personnaliser un connecteur de passerelle 514
Comment Amazon ECS sélectionne les passerelles compatibles 516
Comprendre le déploiement des services 518
Vérifiez la connexion et recherchez l'URL du service 519
Migrer un service 520
Déconnecter un service des passerelles 520
Surveiller une passerelle 521
Afficher l'état de la passerelle 522
Inspecter l'infrastructure gérée (vue de la plateforme) 525
Surveillez les itinéraires des passerelles 525
Surveillez le trafic et l'état de santé des cibles 527
Afficher les journaux de trafic et d'exploitation 530
Résoudre les problèmes liés à une passerelle 531

Défaillances de connexion et de déploiement des services 532
Défaillances de provisionnement des itinéraires de passerelle 535
Problèmes de capacité et de quotas 536
Problèmes liés à l'IAM et aux autorisations 538
Connectivité des services et santé de la cible 539
Échecs de mise à jour 541
Problèmes de dissociation et de suppression 541
Modifications apportées en dehors d'Amazon ECS 542
Définitions de tâches 544
États de définition de tâche 545
Ressources Amazon ECS pouvant bloquer une suppression 546
Architecture de votre application. 547
Pratiques exemplaires en matière de taille des tâches 549
Mise en réseau des tâches pour les instances gérées Amazon ECS 550
Mise en réseau des tâches pour EC2 556
Mise en réseau des tâches pour Fargate 572
Options de stockage pour les tâches 580
Gestion de l’espace mémoire d’échange de conteneur 687
Différences de définition des tâches pour les instances gérées Amazon ECS 688
Différences de définition de tâche pour Fargate 690
Différences entre les définitions de tâche pour les instances EC2 exécutant Windows 700
Création d'une définition de tâche à l'aide de la console 702
Validation JSON 702
CloudFormation piles 702
Procédure 703
Utilisation d’Amazon Q Developer pour proposer des recommandations de définition de tâche dans la console Amazon ECS 735
Conditions préalables 736
Procédure 736
Mise à jour d'une définition de tâche à l'aide de la console 738
Validation JSON 739
Procédure 739
Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console 740
CloudFormation piles 702
Procédure 741
Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console 742

Ressources Amazon ECS pouvant bloquer une suppression ... 546
Procédure ... 743
Cas d'utilisation de définition de tâche ... 744
Définitions de tâches pour les charges de travail GPU ... 744
Définitions de tâche pour les charges de travail de transcodage vidéo ... 769
Définitions de tâches pour les charges de travail de machine learning AWS Neuron ... 781
Définitions de tâche pour les instances de deep learning ... 792
Définitions de tâche pour les charges de travail ARM 64 bits ... 794
Envoyer les journaux à CloudWatch ... 797
Envoyer les journaux à un AWS service ou AWS Partner ... 801
En utilisant des produits non AWS images de conteneurs ... 829
Redémarrer des conteneurs individuels dans les tâches ... 833
Transmission de données sensibles vers un conteneur ... 836
Paramètres de définition de tâche pour les instances gérées Amazon ECS ... 860
Family ... 861
Capacity ... 861
Rôle de tâche ... 861
Rôle d'exécution de tâche ... 862
Mode réseau ... 862
Plateforme d'exécution ... 863
Taille de la tâche ... 864
Autres paramètres de définition de tâche ... 865
Définitions de conteneur ... 872
Paramètres de définition des tâches pour Fargate ... 919
Family ... 919
Capacity ... 919
Rôle de tâche ... 920
Rôle d'exécution de tâche ... 920
Mode réseau ... 921
Plateforme d'exécution ... 921
Taille de la tâche ... 922
Définitions de conteneur ... 926
Nom de l'accélérateur Elastic Inference (obsolète) ... 973
Configuration du proxy ... 974
Volumes ... 976
Étiquettes ... 983

Autres paramètres de définition de tâche 983
Paramètres de définition de tâche pour EC2 986
Family 987
Capacity 987
Rôle de tâche 987
Rôle d'exécution de tâche 988
Mode réseau 988
Plateforme d'exécution 990
Taille de la tâche 991
Définitions de conteneur 992
Nom de l'accélérateur Elastic Inference (obsolète) 1045
Contraintes de placement des tâches 1046
Configuration du proxy 1046
Volumes 1048
Étiquettes 1055
Autres paramètres de définition de tâche 1056
Modèle de définition de tâche 1058
Exemples de définitions de tâches 1069
Serveur Web 1070
`splunk` pilote de journal 1072
`couramment` pilote de journal 1073
`gelf` pilote de journal 1074
Charges de travail sur les instances externes 1074
Image Amazon ECR et rôle IAM de définition de tâche 1076
Point d’entrée avec commande 1076
Dépendances du conteneur 1077
Volumes dans les définitions de tâches 1079
Exemples de définitions de tâche Windows 1079
Clusters 1081
Sélection du type de cluster approprié 1081
Composants d’un cluster 1082
Concepts de cluster 1083
Fournisseurs de capacité 1084
Clusters pour les instances gérées Amazon ECS 1086
Création d’un cluster pour les instances gérées Amazon ECS 1087
Mise à jour d’un cluster pour l’utilisation des instances gérées Amazon ECS 1091

Fournisseurs de capacité d'instances gérées ... 1094
Protection évolutive des tâches ... 1104
Clusters pour Fargate ... 1112
Avis de résiliation Spot Fargate ... 1114
Création d'un cluster pour les charges de travail Fargate ... 1116
Fournisseurs de capacité Amazon ECS pour les charges de travail EC2 ... 1118
Sécurité d'instance de conteneur EC2 ... 1121
Création d'un cluster pour les charges de travail Amazon EC2 ... 1121
Cluster Auto Scaling ... 1128
Instances de conteneur Amazon EC2 ... 1167
Clusters pour instances externes ... 1349
Systèmes d'exploitation et architectures système pris en charge ... 1349
Considérations ... 1350
Création d'un cluster pour les charges de travail d'instance externe ... 1354
Enregistrement d'une instance externe dans un cluster Amazon ECS ... 1356
Annuler l'enregistrement d'une instance externe ... 1362
Mettre à jour le AWS Systems Manager agent et agent de conteneur Amazon ECS ... 1368
Mise à jour d'un cluster ... 1373
Suppression d'un cluster ... 1375
Annulation de l'enregistrement d'une instance de conteneur ... 1376
Procédure ... 1377
Drainage des instances de conteneur ... 1378
Comportement de drainage pour les services ... 1378
Comportement de drainage pour les tâches autonomes ... 1379
Comportement du drainage pour les instances gérées Amazon ECS ... 1160
Procédure ... 1381
Modification du type ou de la taille de l'instance en dehors du groupe Auto Scaling ... 1381
Instances de conteneur Amazon EC2 ... 1382
Cycle de vie ... 1383
Prise en charge de Docker ... 1384
Amazon ECS-optimized AMI ... 1384
Informations supplémentaires ... 1384
Configuration de l'agent de conteneur ... 1384
Installation de l'agent de conteneur Amazon ECS ... 1388
Paramètres de configuration du journal de l'agent de conteneur ... 1394
Configuration des instances de conteneur pour les images Docker privées ... 1395

Nettoyage des tâches et des images 1400
Planification de vos conteneurs 1403
Options de calcul 1405
Cycle de vie d'une tâche 1406
États de cycle de vie 1407
Comment Amazon ECS place les tâches sur des instances de conteneur 1409
Instances gérées Amazon ECS 1409
EC2 1410
Fargate 1411
Autoscaling des instances gérées Amazon ECS et placement des tâches 1411
Utilisation de stratégies pour définir le placement des tâches 1415
Tâches liées au groupe 1421
Définition des instances de conteneur utilisées pour les tâches 1422
Tâches autonomes 1434
Flux de travail des tâches 1434
Exécution d’une application en tant que tâche 1435
À l'aide d'Amazon EventBridge Planificateur pour planifier les tâches 1447
Arrêt d'une tâche 1455
Services 1456
Option de calcul d’infrastructure 1458
Mise à l'échelle automatique des services 1458
Équilibrage de charge des services 1459
Services d’interconnexion 1460
Contrôleurs et stratégies de déploiement de service 1461
Déploiements de services 1718
Révisions de service 1729
Rééquilibrage des zones de disponibilité 1739
Utilisation de l’équilibrage de charge pour répartir le trafic des services 1749
Mise à l'échelle automatique des services 1765
Interconnexion des services 1822
Protection évolutive des tâches 1905
Injection de pannes avec Amazon ECS et Fargate 1915
Mise à jour des paramètres d’un service Amazon ECS 1924
Suppression d'un service 1958
Migration d’un ARN de service court 1959
Logique de limitation du service 1965

Paramètres de définition de service 1967
Modèle de définition de service 1998
Balisage de ressources 2006
Comment les ressources sont étiquetées 2007
Balisage des ressources à la création 2010
Restrictions 2011
ECS-managed Balises Amazon 2012
Mise à jour des balises pour la facturation 2013
Ajout de balises à des ressources 2014
Balises pour les instances gérées Amazon ECS 2016
Tags ajoutés par AWS 2016
Balises personnalisées 2017
Ajout de balises à une instance de conteneur EC2 2018
Ajout de balises à une instance de conteneur externe 2020
Rapports d'utilisation 2021
Task-level coût et utilisation 2022
Contrôle 2024
Pratiques exemplaires en matière de surveillance Amazon ECS 2025
Outils de surveillance 2025
Outils automatiques 2025
Outils manuels 2027
Surveillez Amazon ECS à l'aide CloudWatch 2029
Considérations 2029
Métriques recommandées 2030
Affichage des métriques Amazon ECS 2030
Amazon ECS CloudWatch indicateurs 2032
AWS Fargate métriques d’utilisation 2069
Métriques de réserve de cluster Amazon ECS 2071
Métriques d’utilisation de cluster Amazon ECS 2073
Métriques d’utilisation d’un service Amazon ECS 2075
Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge 2080
Événements Amazon ECS 2081
Gestion des événements 2112
Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée 2116
Métriques supplémentaires pour les instances gérées Amazon ECS 2117

Surveillance des instances gérées Amazon ECS 2122
Surveillance Container Insights 2122
Surveillance des GPU 2123
Surveillance des instances 2123
Détermination de l'état des tâches à l'aide de la surveillance de l'état des conteneurs 2125
Comment l'état des tâches est déterminé 2127
Surveillances de l'état et déconnexions d'agents 2128
Affichage de l'état du conteneur 2128
Surveillance de l'état des instances de conteneurs Amazon ECS 2129
Problèmes liés à l'état des instances de conteneur 2130
Identification des opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications 2131
Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec AWS X-Ray 2131
Spécifier le AWS Distro pour OpenTelemetry sidecar pour AWS X-Ray intégration dans votre définition de tâches 2132
Corrélation des performances des applications Amazon ECS à l'aide des métriques d'application 2133
Exporter les métriques des applications vers Amazon CloudWatch 2134
Exportation des métriques d'application vers Amazon Managed Service for Prometheus ... 2138
Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail 2142
Événements de gestion Amazon ECS dans CloudTrail 2144
Événements liés aux données Amazon ECS dans CloudTrail 2144
Exemples d'événements Amazon ECS 2145
Visualisation CloudWatch Logs Live Tail pour les services et tâches Amazon ECS 2147
Autorisations requises 2148
Procédure 2148
AWS CLI references 2149
Surveillez les opérations d'Amazon ECS avec Action Logs 2149
Présentation d'Amazon ECS Action Logs 2149
Avantages des journaux d'action 2150
Ce qui est enregistré 2150
Comment fonctionnent les journaux d'action 2150
Présentation du schéma de journalisation 2151
Destinations et organisation des journaux 2151
Tarification 2152

Démarrez avec Amazon ECS Action Logs 2152
Résolution des problèmes liés aux journaux d'action Amazon ECS 2156
Surveillance des applications à l’aide des métadonnées 2161
Variables d’environnement 2162
Fichier de métadonnées de conteneur 2163
Métadonnées de tâche disponibles pour les tâches sur les instances gérées 2170
Métadonnées de tâches disponibles pour les tâches sur EC2 2182
Métadonnées de tâches disponibles pour les tâches sur Fargate 2228
Introspection de conteneur 2252
Identification des comportements non autorisés grâce à la surveillance d’exécution 2255
Comment fonctionne la surveillance d’exécution avec Amazon◦ECS 2256
Considérations 2257
Utilisation des ressources 2257
Surveillance d’exécution pour les charges de travail Fargate 2258
Surveillance de l’exécution des charges de travail EC2 2262
Résolution des problèmes de surveillance d’exécution 2267
Surveillance des conteneurs Amazon ECS avec ECS Exec 2270
Considérations 2271
Architecture 2274
Configuration d’ECS Exec 2274
Journalisation à l’aide d’ECS Exec 2277
Utilisation de stratégies IAM pour limiter l'accès à ECS Exec 2281
Résolution des problèmes ECS Exec 2285
Exécution de commandes avec ECS Exec 2285
Recommandations Compute Optimizer 2288
Recommandations relatives à la taille des tâches pour Fargate 2288
Serveur Amazon ECS MCP 2289
Présentation de 2289
Prise en main 2290
Prise en main 2290
Conditions préalables 2291
Configuration 2291
Configurations courantes et meilleures pratiques 2297
Configurations des outils 2299
Configurations des outils 2299
Variables d’environnement 2299

Arguments 2299
Outils 2300
Résolution des problèmes 2307
Résoudre les erreurs liées aux tâches arrêtées 2311
Mises à jour des messages d'erreurs liées aux tâches arrêtées 2312
Affichage des erreurs liées aux tâches arrêtées 2317
Messages d'erreurs liées aux tâches arrêtées 2319
Vérification de la connectivité de la tâche 2341
Affichage des requêtes de rôle IAM 2346
Affichage des messages d'événements de service 2347
Messages d'événement de service Amazon ECS 2348
Messages d'événement relatifs à un dysfonctionnement d'un service Amazon ECS 2362
Messages d'événement de rééquilibrage de service entre zones de disponibilité Amazon ECS 2363
Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS ... 2365
Résolution des problèmes d'autoscaling du service dans Amazon ECS 2368
Capture d'événements 2368
Comment ça marche 2368
Types d'événements 2369
Configuration du groupe de journaux 2369
Considérations 2370
Event-based dépannage 2370
Activer la capture d'événements pour un cluster existant 2371
Affichage des événements de modification de l'état des services et des tâches 2372
Résolution des erreurs d'UC ou de mémoire non valides liées à la définition des tâches 2374
Affichage du journal d'agent de conteneur 2377
Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS 2378
Introspection d'agent 2381
Diagnostic Docker dans Amazon ECS 2383
Recensement des conteneurs Docker dans Amazon ECS 2383
Affichage des journaux Docker dans Amazon ECS 2384
Inspection des conteneurs Docker dans Amazon ECS 2385
Configuration de la sortie détaillée du démon Docker dans Amazon ECS 2386
Résoudre les problèmes liés au Docker `Erreur d'API (500) : devmapper` dans Amazon ECS 2388
Résolution des problèmes liés à ECS Exec 2390

Vérification à l'aide d'Exec Checker 2390
Erreur lors de l'appel `Exécuter la commande` 2390
Résolution des problèmes liés à Amazon ECS Anywhere 2390
Problèmes d'enregistrement d'instance externe 2391
Problèmes de réseau d'instance externe 2392
Problèmes d'exécution de tâches 2392
Résolution des problèmes de chargement des classes Java sur Fargate 2392
Symptômes 2393
Causes 2393
Résolution 2393
Prévention 2394
AWS Fargate limitation des quotas 2395
Limitation de l'API `RunTask` dans Fargate 2396
Ajustement des quotas de taux dans Fargate 2397
Résolution des problèmes liés aux instances gérées Amazon ECS 2397
Conditions préalables 2397
Scénarios courants de résolution des problèmes 2397
Analyse des journaux dans les instances gérées Amazon ECS 2404
Utilisez l'EC2 AWS CLI pour obtenir la sortie de console d'une instance gérée Amazon ECS 2405
Nettoyage 2405
Ressources supplémentaires 2319
Résolution des problèmes liés aux instances gérées Amazon ECS 2405
Erreurs liées aux instances gérées Amazon ECS 2410
Motifs d'échec d'API 2415
Résolution des problèmes avec Amazon Q Developer 2426
Autorisations requises 2427
Procédure 2427
Considérations 2428
En savoir plus 2429
Sécurité 2430
Gestion de l'identité et des accès 2431
Public ciblé 2431
Authentification par des identités 2432
Gestion de l'accès à l'aide de politiques 2433
Fonctionnement d'Amazon Elastic Container Service avec IAM 2435

Identity-based exemples de politiques 2446
AWS politiques gérées pour Amazon ECS 2458
Utilisation des rôles liés à un service 2483
Rôles IAM pour Amazon ECS 2491
Autorisations requises pour la console Amazon ECS 2564
Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements 2575
Autorisations IAM requises pour l'autoscaling d'un service Amazon ECS 2576
Baliser des ressources pendant la création 2577
Résolution des problèmes 2582
Bonnes pratiques IAM 2585
Journalisation et surveillance 2587
Validation de conformité 2589
Pratiques exemplaires en matière de conformité et de sécurité 2590
Conformité d'AWS Fargate FIPS-140 2592
AWS Fargate FIPS-140 Considérations 2592
Utilisation de FIPS sur Fargate 2593
Utiliser CloudTrail pour Fargate FIPS-140 audit 2593
Sécurité de l'infrastructure 2595
Points de terminaison d'un VPC d'interface (AWS PrivateLink) 2595
Modèle de responsabilité partagée 2603
Fargate 2604
EC2 2605
Responsabilité partagée des instances gérées Amazon ECS 2606
AWS responsabilités 2606
Responsabilités client 2607
Pratiques exemplaires en matière de sécurité réseau 2608
Chiffrement en transit 2608
Mise en réseau des tâches 2609
AWS PrivateLink et Amazon ECS 2610
Paramètres d'agent de conteneur 2612
Recommandations de sécurité réseau 2612
Pratiques exemplaires de sécurité des tâches et des conteneurs 2614
Créez un minimum d'images ou utilisez des images sans distribution 2614
Analysez vos images pour détecter les vulnérabilités 2615
Supprimez les autorisations spéciales associées à vos images 2617

Créez un ensemble d'images sélectionnées 2617
Analysez les packages d'applications et les bibliothèques pour détecter les vulnérabilités . 2615
Effectuez une analyse de code statique 2618
Exécutez les conteneurs en tant qu'utilisateur non root 2618
Utilisez un système de fichiers racine en lecture seule 2619
Configurez des tâches avec des limites de processeur et de mémoire (Amazon EC2) 2619
Utilisez des balises immuables avec Amazon ECR 2620
Évitez d'exécuter des conteneurs dotés de privilèges (Amazon EC2) 2620
Supprimez les fonctionnalités Linux inutiles du conteneur 2620
Utilisez une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR 2621
Didacticiels 2622
Création d'une tâche Linux pour Fargate à l'aide du AWS CLI 2624
Conditions préalables 2625
Étape 1 : Créer un cluster 2626
Étape 2 : Enregistrer une définition de tâche Linux 2627
Étape 3 : Répertorier les définitions de tâche 2628
Étape 4 : Créer un service 2629
Étape 5 : Répertorier les services 2630
Étape 6 : Décrire le service en cours d'exécution 2630
Étape 7 : Test 2633
Étape 8 : Nettoyer 2636
Création d'une tâche Windows pour le Fargate à l'aide du AWS CLI 2637
Conditions préalables 2637
Étape 1 : Créer un cluster 2638
Étape 2 : Enregistrement d'une définition de tâche Windows 2639
Étape 3 : Répertorier les définitions de tâche 2640
Étape 4 : Créer un service 2641
Étape 5 : Répertorier les services 2641
Étape 6 : Décrire le service en cours d'exécution 2642
Étape 7 : Nettoyer 2644
Création d'une tâche pour EC2 à l'aide du AWS CLI 2644
Conditions préalables 2645
Créer un cluster 2646
Lancement d’une instance de conteneur avec l’AMI Amazon ECS 2646
Liste des instances de conteneur 2649

Description de votre instance de conteneur 2649
Enregistrement d’une définition de tâche 2652
Liste des définitions de tâche 2654
Créer un service 2655
Liste des services 2656
Description des services 2656
Description de la tâche en cours d’exécution 2657
Test du service Web 2658
nettoyer des ressources ; 2659
Configuration d'Amazon ECS pour écouter CloudWatch les événements 2661
Prérequis : Configuration d'un cluster test 2661
Étape 1 : Créer la fonction Lambda 2661
Étape 2 : Enregistrer une règle d'événement 2662
Étape 3 : Créer une définition de tâche 2663
Étape 4 : Test de la règle 2664
Envoi d’alertes Amazon Simple Notification Service pour les événements de tâche arrêtés 2665
Prérequis : Configuration d'un cluster test 2665
Prérequis : configurer les autorisations pour Amazon SNS 2665
Étape 1 : Créer une rubrique Amazon SNS et s'y abonner 2666
Étape 2 : Enregistrer une règle d'événement 2666
Étape 3 : Test de la règle 2668
Concaténation des messages du journal multiligne ou de suivi de pile 2669
Autorisations IAM requises 2669
Déterminez à quel moment utiliser le paramètre de journal multiligne 2670
Options d'analyse et de concaténation 2671
Déploiement de Fluent Bit sur des conteneurs Windows 2691
Conditions préalables 2694
Étape 1 : Créer les rôles IAM d'accès 2694
Étape 2 : Créer une instance de conteneur Windows Amazon ECS 2695
Étape 3 : Configurer Fluent Bit 2697
Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch 2699
Étape 5 : Exécuter la définition de `tâche ecs-windows-fluent-bit` en tant que service Amazon ECS à l'aide de la stratégie de planification des démons 2701
Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux 2702
Étape 7 : Exécuter la définition de la tâche `windows-app-task` 2703

Étape 8 : Vérifiez les connexions CloudWatch 2704
Étape 9 : nettoyer 2705
Utilisation de gMSA pour les conteneurs Linux EC2 2706
Considérations 2707
Conditions préalables 2708
Configuration 2709
CredSpec file 2716
Utilisation gMSA for Linux conteneurs sur Fargate 2717
Considérations 2718
Conditions préalables 2718
Configuration 2719
CredSpec file 2722
Utilisation de conteneurs Windows et de conteneurs sans domaine à gMSA l'aide du AWS CLI 2723
Conditions préalables 2724
Étape 1 : Création et configuration du gMSA compte sur les services de domaine Active Directory (AD DS) 2726
Étape 2 : charger les informations d'identification dans Secrets Manager 2728
Étape 3 : Modifiez votre CredSpec JSON à inclure sans domaine gMSA Informations 2729
Étape 4 : Téléverser CredSpec vers Amazon S3 2731
Étape 5 (facultative) : créer un cluster Amazon ECS 2731
Étape 6 : créer un rôle IAM pour les instances de conteneur 2732
Étape 7 : créer un rôle d'exécution de tâche personnalisé 2732
Étape 8 : créer un rôle de tâche pour Amazon ECS Exec 2733
Étape 9 : enregistrer une définition de tâche 2735
Étape 10 : enregistrer une instance de conteneur Windows 2737
Étape 11 : vérifier l'instance de conteneur 2737
Étape 12 : exécuter une tâche Windows 2739
Étape 13 : Vérifiez que le conteneur contient gMSA informations d’identification 2739
Étape 14 : nettoyer 2740
Débogage 2741
Utilisation des gMSA les gMSA pour les conteneurs Windows EC2 2742
Considérations 2743
Conditions préalables 2744
Configuration 2745
Utilisation d'Image Builder pour créer des ECS-optimized AMI Amazon personnalisées 2751

Nettoyez l' ECS-optimized AMI Amazon 2753
Utilisation de l’ARN d’image avec l’infrastructure en tant que code (IaC) 2754
Utilisation de l'ARN de l'image avec CloudFormation 2756
Utilisation de l’ARN d’image avec Terraform 2757
Utilisation de AWS Deep Learning Containers 2758
Tutoriel : Création d'un service à l'aide d'un Blue/Green déploiement 2758
Conditions préalables 2759
Étape 1 : Créer un Application Load Balancer 2759
Étape 2 : créer un cluster Amazon ECS 2761
Étape 3 : Enregistrer une définition de tâche 2761
Étape 4 : créer un service Amazon ECS 2762
Étape 5 : Création du AWS CodeDeploy Ressources 2763
Étape 6 : créer et surveiller un CodeDeploy déploiement 2766
Étape 7 : Nettoyer 2770
Service Quotas 2772
Gestion de vos quotas de service dans Console de gestion AWS 2772
Gestion des quotas de service et des seuils de limitation d’API 2774
Elastic Load Balancing 266
Interfaces réseau Elastic 267
AWS Cloud Map 269
Utilisation de l'API pour Amazon ECS 2780
Actions effectuées par la ressource Amazon ECS 2780
Actions de configuration du compte 2781
Actions groupées 2781
Actions des fournisseurs de capacités du cluster 2781
Actions relatives aux conteneurs 2782
Actions des agents de conteneurs 2782
Actions relatives aux instances de conteneurs 2782
Actions relatives à l'espace de noms 2782
Actions de service 2783
Actions de déploiement de services 2783
Actions de révision des services 2783
Actions de définition de tâches 2783
Actions relatives aux tâches 2784
Actions relatives à l'ensemble de tâches 2784
Marquer les actions 2784

Limitation des demandes d’API 2785
Comment l'étranglement est-il appliqué 2785
Limiter les quotas 2786
Ajustement des quotas de limitation des API 2791
Gestion de la limitation des API 2791
Garantir l'idempotence 2792
L'idempotence dans Amazon ECS 2793
Idempotence pour RunTask 2794
Exemples 2795
Recommandations de nouvelle tentative pour les requêtes idempotentes 2798
Historique du document 2800
..... mmdccclvii

Qu'est-ce qu'Amazon Elastic Container Service ?

> **Tip**
>
> Participez aux prochains ateliers pratiques pour découvrir les meilleures pratiques pour Amazon ECS et AWS Fargate. Inscrivez-vous à un atelier.

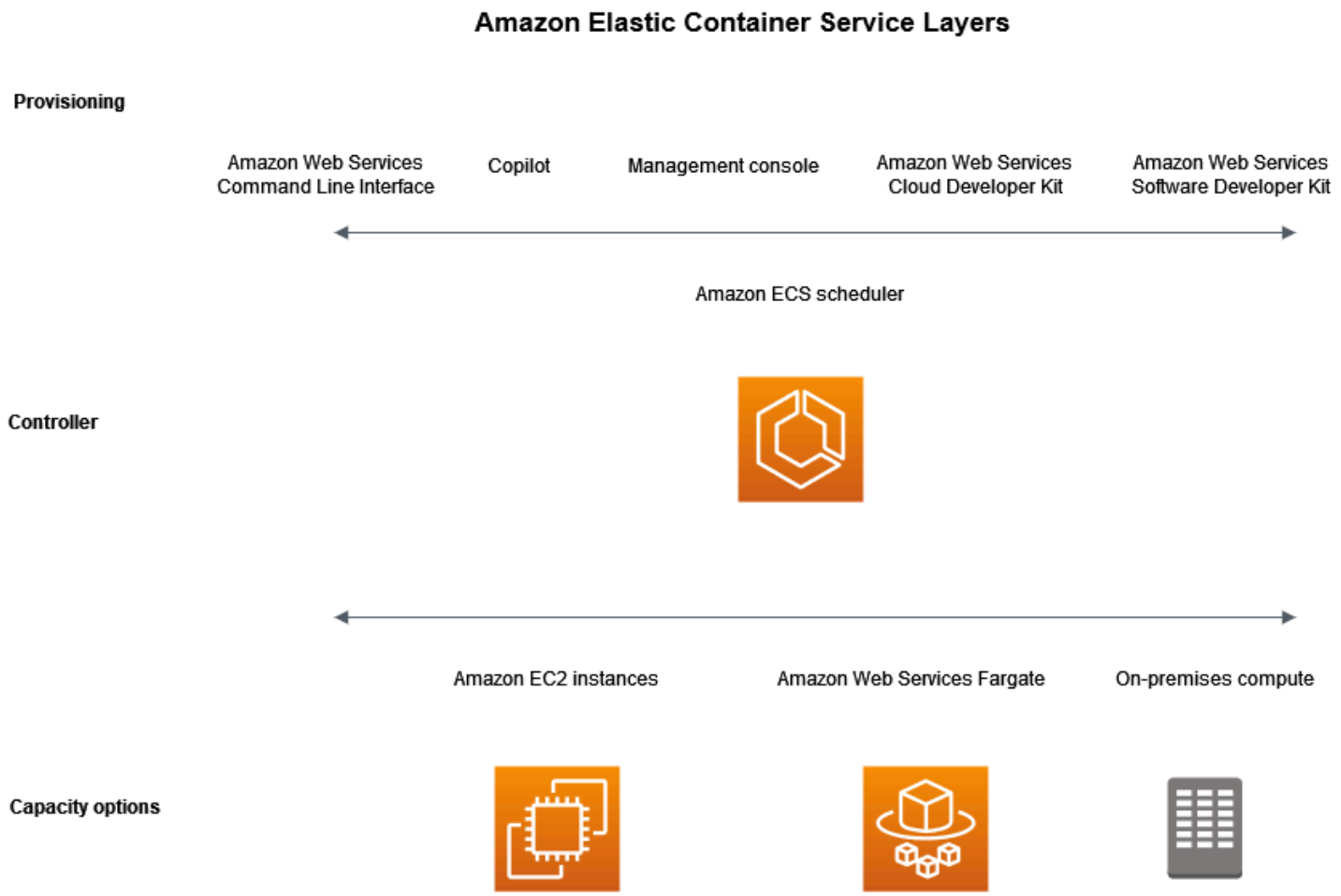
Amazon Elastic Container Service (Amazon ECS) est un service d'orchestration de conteneurs entièrement géré qui vous permet de déployer, de gérer et de dimensionner aisément des applications conteneurisées. En tant que service entièrement géré, Amazon ECS est livré avec AWS meilleures pratiques opérationnelles et de configuration intégrées. Il est intégré à la fois AWS des outils, tels qu'Amazon Elastic Container Registry, et des outils tiers, tels que Docker. Cette intégration permet aux équipes de se concentrer plus facilement sur la création des applications, et non sur l'environnement. Vous pouvez gérer et dimensionner vos charges de travail de conteneurs sur Régions AWS dans le cloud et sur site, sans la complexité de la gestion d'un plan de contrôle.

Terminologie et composants

Amazon ECS comporte trois couches :

- Capacité : l'infrastructure dans laquelle vos conteneurs sont exécutés
- Contrôleur : déploiement et gestion de vos applications qui s'exécutent sur les conteneurs
- Provisionnement : les outils que vous pouvez utiliser pour interagir avec le planificateur afin de déployer et de gérer vos applications et vos conteneurs

Le schéma suivant illustre les couches Amazon ECS.



La capacité correspond à l'infrastructure sur laquelle vos conteneurs s'exécutent. Voici un aperçu des options de capacité :

- Les instances gérées Amazon ECS sont une option de calcul pour Amazon ECS qui vous permet d'exécuter des charges de travail conteneurisées sur différents types d'instances Amazon EC2 tout en confiant la gestion de l'infrastructure à AWS. Avec Amazon ECS Managed Instances, vous pouvez accéder à des fonctionnalités de calcul spécifiques telles que l'accélération GPU, des architectures de processeurs spécifiques, des performances réseau élevées et des types d'instances spécialisés, tout en AWS gère le provisionnement, l'application de correctifs, la mise à l'échelle et la maintenance de l'infrastructure sous-jacente.
- Instances Amazon EC2 dans le AWS nuage

  Vous choisissez le type d'instance, le nombre d'instances et gérez la capacité.
- Sans serveur dans AWS nuage

Fargate est un moteur de calcul sans serveur avec paiement à l'utilisation. Avec Fargate, vous n'avez pas besoin de gérer les serveurs, de gérer la planification de la capacité ou d'isoler les charges de travail des conteneurs pour des raisons de sécurité.

- On-premises machines virtuelles (VM) ou serveurs

  Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS.

Le planificateur Amazon ECS est le logiciel qui gère vos applications.

Caractéristiques

Amazon ECS offre les fonctionnalités de haut niveau suivantes :

Définition de tâche

Le plan de l’application.

Cluster

L’infrastructure sur laquelle votre application s’exécute.

Sous-tâche

Application telle qu’un traitement par lots qui exécute un travail, puis s’arrête.

Service

Une application sans état à exécution longue durée.

Paramètres du compte

Autorise l’accès aux fonctionnalités.

Cluster Auto Scaling

Amazon ECS gère la mise à l’échelle des instances Amazon EC2 enregistrées sur votre cluster.

Service Auto Scaling

Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service.

Allouer

Il existe plusieurs options pour le provisionnement d'Amazon ECS :

- Console de gestion AWS — Offre une interface web que vous pouvez utiliser pour accéder à vos ressources Amazon ECS.
- AWS Command Line Interface (AWS CLI) — Fournit des commandes pour un large éventail de AWS services, notamment Amazon ECS. Elle est prise en charge sur Windows, Mac et Linux. Pour plus d'informations, consultez AWS Command Line Interface.
- AWS SDK — Fournit des API spécifiques à chaque langue et prend en charge de nombreux détails de connexion. Ces outils incluent le calcul des signatures, la gestion des nouvelles tentatives de demande et la gestion des erreurs. Pour plus d'informations, consultez Kits SDK AWS.
- AWS CDK— Fournit un cadre de développement logiciel open source que vous pouvez utiliser pour modéliser et allouer vos ressources d'applications cloud à l'aide de langages de programmation familiers. Le AWS CDK alloue vos ressources de manière sûre et répétable grâce à AWS CloudFormation.

Tarification

La tarification d'Amazon ECS dépend de l'option de capacité que vous choisissez pour vos conteneurs.

- Tarification Amazon ECS : informations sur la tarification pour Amazon ECS.
- AWS Fargate tarification — Informations sur les prix de Fargate.

Services connexes

Services à utiliser avec Amazon ECS

Vous pouvez utiliser d'autres Services AWS pour vous aider à déployer vos tâches et services sur Amazon ECS.

Amazon EC2 Auto Scaling

> Permet de s'assurer que vous disposez du nombre adéquat d'instances Amazon EC2 pour gérer la charge de votre application.

Amazon CloudWatch

Surveillance de vos services et vos tâches.

Amazon Elastic Container Registry

Transmission et gestion des images de conteneur.

Elastic Load Balancing

Répartition automatique du trafic de service entrant.

Amazon GuardDuty

Détection des utilisations potentiellement non autorisées ou malveillantes de vos instances de conteneurs et de vos charges de travail.

Création et utilisation de ressources Amazon ECS

Les guides suivants fournissent une introduction aux outils disponibles pour accéder à Amazon ECS, ainsi que des procédures d'introduction pour exécuter des conteneurs. Docker Basics vous permet d'effectuer les étapes de base pour créer une image de conteneur Docker et la charger dans un référentiel privé Amazon ECR. Les guides de démarrage vous expliqueront comment utiliser AWS L'interface de ligne de commande Copilot et le Console de gestion AWS pour effectuer les tâches courantes liées à l'exploitation de vos conteneurs sur Amazon ECS et AWS Fargate.

Table des matières

- Configurer l'utilisation d'Amazon ECS
- Création d'une image de conteneur à utiliser sur Amazon ECS
- Création d'une tâche pour les instances gérées Amazon ECS
- Apprenez à créer une tâche pour les instances gérées Amazon ECS à l'aide du AWS CLI
- Création d'une tâche Amazon ECS Linux pour Fargate
- Création d'une tâche Windows Amazon ECS pour Fargate
- Création d'une tâche Windows Amazon ECS pour EC2
- Création de ressources Amazon ECS à l'aide du AWS CDK
- Création de ressources Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot

Configurer l'utilisation d'Amazon ECS

Si vous êtes déjà inscrit à Amazon Web Services (AWS) et que vous avez utilisé Amazon Elastic Compute Cloud (Amazon EC2), vous pouvez presque utiliser Amazon ECS. Le processus de configuration est très similaire pour les deux services. Le guide suivant vous prépare au lancement de votre premier cluster Amazon ECS.

Effectuez les tâches suivantes pour vous préparer à utiliser Amazon ECS.

Console de gestion AWS

Le Console de gestion AWS est une interface basée sur un navigateur permettant de gérer les ressources Amazon ECS. Elle offre une vue d'ensemble du service, ce qui facilite l'exploration des fonctions et rôles d'Amazon ECS sans avoir besoin d'outils supplémentaires. Une multitude de didacticiels et de démonstrations sont disponibles pour vous guider dans l'utilisation de la console.

Pour obtenir un didacticiel pour vous guider dans la console, consultez Création et utilisation de ressources Amazon ECS.

Au début, de nombreux clients préfèrent utiliser la console, car elle fournit un feedback visuel instantané indiquant si les actions qu'ils entreprennent sont couronnées de succès. AWS clients qui connaissent les Console de gestion AWS, peut facilement gérer les ressources associées telles que les équilibreurs de charge et les instances Amazon EC2.

Commencez par Console de gestion AWS.

Inscrivez-vous pour Compte AWS

Pour commencer AWS, vous avez besoin d'un Compte AWS. Il existe deux manières de s'inscrire AWS. Si vous souhaitez contrôler totalement la configuration de votre compte et l'emplacement de stockage de votre contenu, ou si vous avez des charges de travail réglementées telles que HIPAA ou FedRAMP, utilisez Inscrivez-vous pour AWS (avancé). Si vous êtes un nouvel utilisateur et que vous souhaitez utiliser un identifiant que vous possédez déjà pour vous connecter AWS et créez un AWS environnement avec des valeurs par défaut préconfigurées dans select Régions AWS, utiliser Inscrivez-vous pour AWS.

Créer un Virtual Private Cloud

Vous pouvez utiliser Amazon Virtual Private Cloud (Amazon VPC) pour lancer AWS ressources dans un réseau virtuel que vous avez défini. Nous vous recommandons de lancer vos instances de conteneurs dans un VPC.

Si vous avez un VPC par défaut, vous pouvez ignorer cette section et passer à la tâche suivante, Création d’un groupe de sécurité. Pour déterminer si vous disposez d’un VPC par défaut, consultez la section Utilisation de votre VPC et de vos sous-réseaux par défaut dans le Guide de l’utilisateur Amazon VPC. Sinon, vous pouvez créer un VPC autre que celui par défaut dans votre compte en suivant les étapes suivantes.

Pour plus d’informations sur la création d’un VPC, consultez la section Création d’un VPC dans le Guide de l’utilisateur Amazon VPC et utilisez le tableau suivant pour déterminer quelles options sélectionner.

Option	Value	
Ressources à créer	VPC uniquement	

Option	Value
Nom	Vous pouvez, si vous le souhaitez, nommer votre VPC.
Bloc d'adresse CIDR IPv4	Entrée manuelle CIDR IPv4 La taille du bloc d'adresse CIDR doit être comprise entre /16 et /28.
Bloc d'adresse CIDR IPv6	Pas de bloc d'adresse CIDR IPv6
Tenancy	Par défaut

Pour plus d'informations sur Amazon VPC, consultez Qu'est-ce qu'Amazon VPC ? dans le Guide de l'utilisateur Amazon VPC.

Création d’un groupe de sécurité

Les groupes de sécurité font office de pare-feu pour les instances de conteneur associées, en contrôlant le trafic entrant et le trafic sortant au niveau de l'instance de conteneur. Vous pouvez ajouter des règles à un groupe de sécurité qui vous permettent de vous connecter à votre instance de conteneur à partir de votre adresse IP avec SSH. Vous pouvez aussi ajouter des règles qui permettent les accès HTTP et HTTPS entrants et sortants depuis n'importe quel emplacement. Ajoutez des règles pour ouvrir les ports qui sont requis par vos tâches. Les instances de conteneur ont besoin d'un accès réseau externe pour communiquer avec le point de terminaison d'Amazon ECS service.

Si vous prévoyez de lancer des instances de conteneur dans plusieurs régions, vous devez créer un groupe de sécurité dans chaque région. Pour plus d’informations, consultez Régions et zones de disponibilité dans le Guide de l’utilisateur Amazon EC2.

> **Tip**
>
> Vous avez besoin de l'adresse IP publique de votre ordinateur local, que vous pouvez obtenir à l'aide d'un service. Par exemple, nous proposons le service suivant : http://checkip.amazonaws.com/ ou https://checkip.amazonaws.com/. Pour trouver un autre

service qui fournit votre adresse IP, utilisez l'expression de recherche « what is my IP address » (quelle est mon adresse IP). Si votre connexion s'effectue via un fournisseur d'accès Internet (FAI) ou derrière un pare-feu sans adresse IP statique, vous devez déterminer la plage d'adresses IP utilisée par les ordinateurs clients.

Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Création d'un groupe de sécurité pour votre instance Amazon EC2 dans le Guide de l'utilisateur Amazon EC2 et utilisez le tableau suivant pour déterminer quelles options sélectionner.

Option	Value
Région	La même région que celle que vous avez créée avec votre paire de clés.
Nom	Un nom facile à retenir, tel que ecs-instances-default-cluster.
VPC	VPC par défaut (marqué « (par défaut) »). **Note** Si votre compte prend en charge Amazon EC2 Classic, sélectionnez le VPC créé lors de l'étape précédente.

Pour obtenir des informations sur les règles de trafic sortant à ajouter pour vos cas d'utilisation, consultez la section Règles de groupe de sécurité pour différents cas d'utilisation dans le Guide de l'utilisateur Amazon EC2.

Les instances de conteneur Amazon ECS ne nécessitent pas que tous les ports entrants soient ouverts. Vous pouvez toutefois ajouter une règle SSH, afin de vous connecter à l'instance de conteneur et d'examiner les tâches avec les commandes Docker. Vous pouvez également ajouter

des règles pour HTTP et HTTPS si vous voulez que votre instance de conteneur héberge une tâche qui exécute un serveur web. Les instances de conteneur ont besoin d'un accès réseau externe pour communiquer avec le point de terminaison d'Amazon ECS service. Appliquez la procédure suivante pour ajouter ces règles de groupe de sécurité facultatives.

Ajoutez les trois règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur le mode de création d'un groupe de sécurité, consultez la section Configuration des règles de groupe de sécurité dans le Guide de l'utilisateur Amazon EC2.

Option	Value	
Règle HTTP	Type : HTTP Source : n'importe où (`0.0.0.0/0`) Cette option ajoute automatiquement la valeur 0.0.0.0/0 Bloc CIDR IPv4 en tant que source. Cette solution est acceptable pour une brève durée dans un environnement de test, mais n'est pas sûre dans des environnements de production. Dans un environnement de production, vous autorisez uniquement l'accès à votre instance pour une adresse IP ou une plage d'adresses spécifiques.	
Règle HTTP	Type : HTTPS Source : n'importe où (`0.0.0.0/0`) Cette solution est acceptable pour une brève durée dans un environnement de test,	

Option	Value	
	mais n'est pas sûre dans des environnements de productio n. Dans un environnement de production, vous autorisez uniquement l'accès à votre instance pour une adresse IP ou une plage d'adresses spécifiques.	

Option	Value
Règle SSH	Type : SSH Source : choisissez Custom (Personnalisée) et spécifiez l'adresse IP publique de votre ordinateur ou réseau en notation CIDR. Pour spécifier une adresse IP individuelle en notation CIDR, ajoutez le préfixe de routage `/32`. Par exemple, si votre adresse IP est `203.0.113.25` , spécifiez `203.0.113.25/32` . Si votre entreprise alloue des adresses à partir d'une plage, spécifiez la plage complète, telle que `203.0.113.0/24` . **Important** Pour des raisons de sécurité, il est déconseillé d'autoriser l'accès de SSH à votre instance à partir de toutes les adresses IP (`0.0.0.0/0`), si ce n'est à titre de test et pour une très brève durée.

Créer les informations d'identification pour vous connecter à votre instance EC2

Pour Amazon ECS, une paire de clés n’est nécessaire que si vous avez l’intention d’utiliser EC2.

AWS utilise la cryptographie à clé publique pour sécuriser les informations de connexion de votre instance. Une instance Linux, comme une instance de conteneur Amazon ECS, n'a aucun mot de passe à utiliser pour l'accès SSH. Une paire de clés vous permet de vous connecter en toute sécurité à votre instance. Vous devez indiquer le nom de la paire de clés au lancement de votre instance de conteneur, puis fournir la clé privée lorsque vous vous connectez avec SSH.

Si vous n'avez pas encore créé de paire de clés, vous pouvez le faire à l'aide de la console Amazon EC2. Si vous prévoyez de lancer des instances dans plusieurs régions, vous devez créer une paire de clés dans chaque région. Pour de plus amples informations sur les régions, consultez la section Régions et zones de disponibilité dans le Guide de l’utilisateur Amazon EC2.

Pour créer une paire de clés

- Utilisez la console Amazon EC2 pour créer une paire de clés. Pour plus d’informations sur la création d’une paire de clés, consultez la section Création d’une paire de clés dans le Guide de l’utilisateur Amazon EC2.

Pour plus d’informations sur la connexion à votre instance, consultez la section Connexion à votre instance Linux dans le Guide de l’utilisateur Amazon EC2.

Installer la AWS CLI

Le Console de gestion AWS peut être utilisé pour gérer toutes les opérations manuellement avec Amazon ECS. Cependant, vous pouvez installer AWS CLI sur votre poste de travail local ou sur un boîtier de développeur afin de créer des scripts capables d'automatiser les tâches de gestion courantes dans Amazon ECS.

Pour utiliser le AWS CLI avec Amazon ECS, installez la dernière version AWS CLI version. Pour plus d'informations sur l'installation du AWS CLI ou en le mettant à niveau vers la dernière version, voir Installation ou mise à jour vers la dernière version du AWS CLI dans le AWS Command Line Interface Guide de l'utilisateur.

Le AWS Command Line Interface (AWS CLI) est un outil unifié que vous pouvez utiliser pour gérer votre AWS services. Avec ce seul outil, vous pouvez à la fois contrôler plusieurs AWS services et

automatisez ces services par le biais de scripts. Les commandes Amazon ECS du AWS CLI sont le reflet de l'API Amazon ECS.

Le AWS CLI convient aux clients qui préfèrent et ont l'habitude de créer des scripts et de s'interfacer avec un outil de ligne de commande et qui savent exactement quelles actions ils souhaitent effectuer sur leurs ressources Amazon ECS. Le AWS CLI est également utile aux clients qui souhaitent se familiariser avec les API Amazon ECS. Les clients peuvent utiliser le AWS CLI pour effectuer un certain nombre d'opérations sur les ressources Amazon ECS, notamment les opérations de création, de lecture, de mise à jour et de suppression, directement depuis l'interface de ligne de commande.

Utilisez le AWS CLI si vous êtes ou souhaitez vous familiariser avec les API Amazon ECS et les commandes CLI correspondantes et si vous souhaitez écrire des scripts automatisés et effectuer des actions spécifiques sur les ressources Amazon ECS.

AWS fournit également les outils de ligne de commande AWS Tools for Windows PowerShell. Pour plus d'informations, consultez le Guide de l'utilisateur AWS Tools for Windows PowerShell.

Prochaines étapes pour l'utilisation d'Amazon ECS

Après avoir installé le AWS CLI, il existe de nombreux outils différents que vous pouvez utiliser tout en continuant à utiliser Amazon ECS. Les liens suivants expliquent ce que sont certains de ces outils et donnent des exemples de leur utilisation avec Amazon ECS.

- Créez votre première image de conteneur avec Docker et transférez-la vers Amazon ECR pour l'utiliser dans vos définitions de tâches Amazon ECS.
- Création d'une tâche Amazon ECS Linux pour Fargate.
- Création d'une tâche Windows Amazon ECS pour Fargate.
- Création d'une tâche Windows Amazon ECS pour EC2.
- À l'aide de votre langage de programmation préféré, définissez l'infrastructure ou l'architecture en tant que code avec Création de ressources Amazon ECS à l'aide du AWS CDK.
- Définissez et gérez tout AWS ressources de votre environnement grâce à un déploiement automatisé à l'aide Utilisation d'Amazon ECS avec AWS CloudFormation.
- Utilisez la version complète Création de ressources Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot flux de travail de développement de bout en bout pour créer, publier et exploiter des applications conteneurs conformes à AWS les meilleures pratiques en matière d'infrastructure.

Création d'une image de conteneur à utiliser sur Amazon ECS

Amazon ECS utilise des images Docker dans les définitions de tâches pour lancer des conteneurs. Docker est une technologie qui fournit les outils nécessaires pour créer, exécuter, tester et déployer des applications distribuées sur des conteneurs.

Amazon ECS planifie les applications conteneurisées sur des instances de conteneur ou sur AWS Fargate. Les applications conteneurisées sont conditionnées sous forme d'images de conteneur. Cet exemple crée une image de conteneur pour un serveur Web.

Vous pouvez créer votre première image Docker, puis la transférer vers Amazon ECR, qui est un registre de conteneurs, afin de l'utiliser dans vos définitions de tâches Amazon ECS. Ce guide part du principe que vous possédez des connaissances élémentaires de Docker et de son fonctionnement. Pour plus d'informations sur Docker, consultez les sections Qu'est-ce que Docker ? et la Documentation Docker.

Conditions préalables

Avant de commencer, assurez-vous de respecter les prérequis suivants :

- Veillez à avoir terminé les étapes de configuration d'Amazon ECR. Pour plus d'informations, consultez la section Déplacement d'une image au cours de son cycle de vie dans Amazon ECR dans le Guide de l'utilisateur Amazon Elastic Container Registry.
- Votre utilisateur dispose des autorisations IAM requises pour accéder au service Amazon ECR et l'utiliser. Pour en savoir plus, consultez Politiques gérées Amazon ECR.
- Vous avez installé Docker. Pour les étapes d'installation de Docker pour Amazon Linux 2023, consultez la section Installation de Docker sur AL2023. Pour tous les autres systèmes d'exploitation, consultez la documentation Docker dans Présentation de Docker Desktop.
- Vous avez AWS CLI installé et configuré. Pour plus d'informations, consultez la section Installation ou mise à jour vers la version la plus récente de l' AWS CLI dans le Guide de l'utilisateur AWS Command Line Interface .

Si vous n'avez pas besoin d'un environnement de développement local et que vous préférez utiliser une instance Amazon EC2 pour utiliser Docker, nous proposons la procédure suivante pour lancer une instance Amazon EC2 à l'aide d'Amazon Linux 2023 et installer Docker Engine et la CLI Docker.

Installation de Docker sur AL2023

Docker est disponible sur plusieurs systèmes d'exploitation, notamment les distributions Linux les plus modernes, comme Ubuntu et même MacOS et Windows. Pour en savoir plus sur la façon d'installer Docker sur votre système d'exploitation, consultez le guide d'installation Docker.

Vous n'avez pas besoin d'un système de développement local pour utiliser Docker. Si vous utilisez déjà Amazon EC2, vous pouvez lancer une instance Amazon Linux 2023 et installer Docker pour démarrer.

Si vous avez déjà installé Docker, passez à Créer une image Docker.

Pour installer Docker sur une instance Amazon EC2 à l'aide d'une AMI Amazon Linux 2023

1. Lancez une instance avec la dernière AMI Amazon Linux 2023. Pour plus d'informations, consultez la section Lancer une instance EC2 à l'aide de l'assistant de lancement d'instance dans la console dans le Guide de l'utilisateur Amazon EC2.
2. Connectez-vous à votre instance. Pour plus d'informations, consultez la section Connexion à votre instance EC2 dans le Guide de l'utilisateur Amazon EC2.
3. Mettez à jour les packages installés et le cache du package sur votre instance.

   ```
   sudo yum update -y
   ```

4. Installez le package de Docker Community Edition le plus récent.

   ```
   sudo yum install docker
   ```

5. Lancez le service Docker.

   ```
   sudo service docker start
   ```

6. Ajoutez le `ec2-user` au groupe `docker` afin de pouvoir exécuter les commandes Docker sans utiliser le `sudo`.

   ```
   sudo usermod -a -G docker ec2-user
   ```

7. Déconnectez-vous et reconnectez-vous pour récupérer les nouvelles autorisations de groupe `docker`. Vous pouvez effectuer ces opérations en fermant votre fenêtre de terminal SSH actuelle et en vous reconnectant à votre instance dans une nouvelle fenêtre. Votre nouvelle session SSH disposera des autorisations de groupe `docker` appropriées.

8. Vérifiez que `ec2-user` peut exécuter les commandes Docker sans `sudo`.

```
docker info
```

> **Note**
>
> Dans certains cas, vous devrez peut-être redémarrer votre instance pour autoriser `ec2-user` à accéder au démon Docker. Essayez de redémarrer l'instance si vous voyez l'erreur suivante :
>
> ```
> Cannot connect to the Docker daemon. Is the docker daemon running on this
>  host?
> ```

Créer une image Docker

Les définitions de tâches Amazon ECS utilisent des images de conteneur pour lancer des conteneurs sur des instances de conteneur dans vos clusters. Dans cette section, vous créez une image Docker pour une simple application web et vous la testez sur votre système local ou l'instance Amazon EC2, puis transférez l'image vers le registre de conteneur Amazon ECR afin de pouvoir l'utiliser dans une définition de tâche Amazon ECS.

Créer une image Docker d'une application web simple

1. Créez un fichier, appelé `Dockerfile`. Un fichier Dockerfile est un manifeste qui décrit l'image de base à utiliser pour votre image Docker et ce que vous voulez installer et exécuter dessus. Pour en savoir plus sur les fichiers Dockerfile, consultez la référence Dockerfile.

```
touch Dockerfile
```

2. Modifiez le `Dockerfile` que vous venez de créer et ajoutez le contenu qui suit.

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest

# Update installed packages and install Apache
RUN yum update -y && \
 yum install -y httpd

# 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
```

Ce fichier Dockerfile utilise l'image publique Amazon Linux 2023 hébergée sur Amazon ECR Public. Les instructions RUN mettent à jour les caches du package, installent certains packages logiciels pour le serveur web et écrivent ensuite le message « Hello World! » contenu à la racine du document des serveurs Web. L'instruction EXPOSE signifie que le port 80 du conteneur est celui qui est à l'écoute, et l'instruction CMD démarre le serveur Web.

3. Créez l'image Docker à partir de votre fichier Dockerfile.

> **Note**
>
> Certaines versions de Docker peuvent nécessiter le chemin complet de votre Dockerfile dans la commande suivante, au lieu du chemin relatif indiqué dans l'exemple suivant. Si vous exécutez la commande sur un système basé sur ARM, tel que Apple Silicon, utilisez l'option --platform « --platform linux/amd64«.

```
docker build -t hello-world .
```

4. Répertoriez l'image de votre conteneur.

```
docker images --filter reference=hello-world
```

Sortie :

```
REPOSITORY          TAG                 IMAGE ID            CREATED
 SIZE
hello-world         latest              e9ffedc8c286        4 minutes ago
 194MB
```

5. Exécutez la nouvelle image. L'option `-p 80:80` mappe le port exposé 80 du conteneur au port 80 du système hôte.

```
docker run -t -i -p 80:80 hello-world
```

> **Note**
>
> La sortie du serveur Web Apache est affichée dans la fenêtre du terminal. Vous pouvez ignorer le message « `Could not reliably determine the fully qualified domain name` ».

6. Ouvrez un navigateur et pointez vers le serveur qui exécute Docker et qui héberge votre conteneur.

   - Si vous utilisez une instance EC2, il s'agit de la valeur de DNS public du serveur, c'est-à-dire la même adresse que celle que vous utilisez pour vous connecter à l'instance avec le protocole SSH. Assurez-vous que le groupe de sécurité de votre instance autorise le trafic entrant sur le port 80.
   - Si vous exécutez Docker localement, pointez votre navigateur sur http://localhost/.

   Vous devriez voir une page web avec « Hello, World! » .

7. Arrêtez le conteneur Docker en appuyant sur Ctrl + c.

Transmettre votre image à Amazon Elastic Container Registry

Amazon ECR est géré AWS service de registre d'images géré. Vous pouvez utiliser la CLI Docker pour transmettre, extraire et gérer les images dans vos référentiels Amazon ECR. Pour consulter des détails sur le produit Amazon ECR, des études de cas de clients et des FAQ, rendez-vous sur les pages détaillées du produit Amazon Elastic Container Registry.

Pour étiqueter votre image et la transmettre à Amazon ECR

1. Créez un référentiel Amazon ECR pour stocker votre image `hello-world`. Notez la valeur `repositoryUri` dans la sortie.

   Substitut `region`, avec votre Région AWS, par exemple, `us-east-1`.

```
aws ecr create-repository --repository-name hello-repository --region region
```

Sortie :

```
{
    "repository": {
        "registryId": "aws_account_id",
        "repositoryName": "hello-repository",
        "repositoryArn": "arn:aws:ecr:region:aws_account_id:repository/hello-
repository",
        "createdAt": 1505337806.0,
        "repositoryUri": "aws_account_id.dkr.ecr.region.amazonaws.com/hello-
repository"
    }
}
```

2. Étiquetez l'image `hello-world` avec la valeur `repositoryUri` de l'étape précédente.

```
docker tag hello-world aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

3. Exécutez la commande aws ecr get-login-password. Spécifiez l'URI du registre sur lequel vous souhaitez vous authentifier. Pour plus d'informations, veuillez consulter Authentification de registre dans le Guide de l'utilisateur Amazon Elastic Container Registry.

```
aws ecr get-login-password --region region | docker login --username AWS --
password-stdin aws_account_id.dkr.ecr.region.amazonaws.com
```

Sortie :

```
Login Succeeded
```

> **Important**
>
> Si vous recevez une erreur, installez la dernière version de l' AWS CLI ou effectuez une mise à niveau vers cette version. Pour plus d'informations, consultez la section Installation ou mise à jour vers la version la plus récente de l' AWS CLI dans le Guide de l'utilisateur AWS Command Line Interface .

4. Transférez l'image dans Amazon ECR avec la valeur `repositoryUri` de l'étape précédente.

```
docker push aws_account_id.dkr.ecr.region.amazonaws.com/hello-repository
```

Nettoyage

Pour continuer à créer une définition de tâche Amazon ECS et à lancer une tâche avec votre image de conteneur, passez à la page Étapes suivantes. Une fois que vous avez fini de tester votre image Amazon ECR, vous pouvez supprimer le référentiel afin qu'aucun stockage d'images ne vous soit facturé.

```
aws ecr delete-repository --repository-name hello-repository --region region --force
```

Étapes suivantes

Vos définitions de tâches nécessitent un rôle d'exécution de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Une fois que vous avez créé et transféré votre image de conteneur vers Amazon ECR, vous pouvez utiliser cette image dans une définition de tâche. Pour plus d'informations, consultez :

- the section called "Création d'une tâche Linux pour Fargate"
- the section called "Création d'une tâche Windows pour Fargate"
- Création d'une tâche Linux Amazon ECS pour le Fargate avec AWS CLI

Création d'une tâche pour les instances gérées Amazon ECS

Découvrez comment utiliser Amazon ECS avec les instances gérées Amazon ECS pour exécuter une application conteneurisée.

Conditions préalables

Avant de commencer le didacticiel, vous devez avoir effectué les opérations suivantes :

- Vous avez suivi les étapes décrites dans la section Configuration pour utiliser Amazon ECS.
- Vous devez avoir suivi les étapes de the section called "Configuration".

- Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :
  - Rôle d'infrastructure : permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

    Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
  - Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

    Le nom du rôle d'instance doit inclure `ecsInstanceRole` comme préfixe pour correspondre à `iam:PassRole` action dans le rôle de l'infrastructure.

    Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Amazon ECR Public. Vos instances doivent donc disposer d'un accès Internet. Pour attribuer à vos instances une route vers Internet, utilisez l'une des options suivantes :
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et attribuez une adresse IP publique aux instances.

  Pour plus d'informations, consultez the section called "Créer un Virtual Private Cloud".

  Pour de plus amples informations sur les règles et les groupes de sécurité, consultez les sections Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l'utilisateur Amazon Virtual Private Cloud.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que AWS CloudShell? dans le AWS CloudShell Guide de l'utilisateur.

Étape 1 : créer un cluster

1. Ouvrez la console Amazon ECS à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, pour Nom du cluster, saisissez un nom unique pour votre cluster.

6. Sous Infrastructure, choisissez Fargate et EC2 géré.
7. Configurez les paramètres des instances gérées :
   a. Pour Rôle d'infrastructure, choisissez le rôle IAM que vous avez créé pour la gestion de l'infrastructure des instances gérées.
   b. Pour Profil d'instance, sélectionnez l'`ecsInstanceRole` que vous avez créé.
   c. Pour Attributs de l'instance, choisissez Utiliser les valeurs ECS par défaut.
8. Sous Mise en réseau, configurez le VPC et les sous-réseaux pour vos instances gérées :
   a. Pour VPC, sélectionnez le VPC qui héberge les instances gérées.
   b. Pour Sous-réseaux, sélectionnez un ou plusieurs sous-réseaux dans lesquels vos instances gérées seront lancées.
   c. Pour Security groups (Groupes de sécurité), sélectionnez un ou plusieurs groupes de sécurité.
9. (Facultatif) Pour vous aider à identifier votre cluster, développez Balises, puis configurez vos balises.
10. Cliquez sur Créer.

Étape 2 : Créer une définition de tâche

Une définition de tâche est le plan de votre application. Chaque fois que vous lancez une tâche dans Amazon ECS, vous spécifiez une définition de tâche. Le service connaît ensuite l'image Docker à utiliser pour les conteneurs, le nombre de conteneurs à utiliser dans la tâche et l'allocation des ressources pour chaque conteneur. Pour créer une définition de tâche, procédez comme suit :

1. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
2. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
3. Copiez et collez le code JSON suivant dans l'éditeur en remplaçant le code JSON prérempli :

   Remplacer **`account-id`** avec votre AWS identifiant de compte et **`region`** avec la région que vous utilisez.

```
{
  "family": "managed-instance-tutorial",
  "networkMode": "awsvpc",
```

```
  "executionRoleArn": "arn:aws:iam::account-id:role/ecsTaskExecutionRole",
  "containerDefinitions": [
    {
      "name": "sample-app",
      "image": "public.ecr.aws/docker/library/httpd:latest",
      "essential": true,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
      ],
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/managed-instance-tutorial",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "ecs"
        }
      }
    }
  ],
  "requiresCompatibilities": [
    "MANAGED_INSTANCES"
  ],
  "cpu": "1024",
  "memory": "2048"
}
```

4. Cliquez sur Créer.

Étape 3 : Création d'un service

Un service Amazon ECS service permet d'exécuter et de gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Suivez ces étapes pour créer un service :

1. Dans le panneau de navigation, choisissez Clusters, puis choisissez le cluster **`managed-instance-tutorial`**.
2. Sous l'onglet Services choisissez Create (Créer).

3. Pour Famille de définitions de tâches, choisissez tutoriel sur les instances gérées.
4. Pour Nom du service, entrez **managed-instance-tutorial-service**.
5. En dessous Environnement, Choisissez Stratégie pour les fournisseurs de capacités.
6. Sous Mise en réseau, configurez les éléments suivants :

    a. Choisissez un VPC existant ou créez-en un nouveau.
    b. Pour Sous-réseaux, choisissez les sous-réseaux à utiliser.
    c. Pour Groupes de sécurité, choisissez un groupe de sécurité existant ou créez-en un nouveau qui autorise le trafic entrant sur le port 80.
7. Cliquez sur Créer.

Étape 4 : Afficher votre service

Une fois votre service lancé, vous pouvez le consulter pour en savoir plus et le tester.

1. Sélectionnez le service **managed-instance-tutorial-service**.
2. À partir du Tâches onglet, choisissez l'ID de tâche de la tâche en cours.
3. Sous Réseau, dans Public IP, sélectionnez Adresse ouverte.
4. Vous devriez voir la page de test du serveur HTTP Apache, qui confirme que le serveur Web fonctionne correctement.

Étape 5 : nettoyer

Une fois ce didacticiel terminé, vous devez nettoyer les ressources associées afin d’éviter tout frais liés à des ressources que vous n’utilisez pas.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster **managed-instance-tutorial**.
3. Choisissez l'onglet Services.
4. Sélectionnez le service **managed-instance-tutorial-service**, puis choisissez Supprimer.
5. À l’invite de confirmation, saisissez **delete** puis choisissez Supprimer.
6. Une fois le service supprimé, choisissez Clusters dans le volet de navigation.
7. Sur la page Clusters, sélectionnez le cluster **managed-instance-tutorial**, puis choisissez Supprimer le cluster.

8. À l'invite de confirmation, saisissez **`delete managed-instance-tutorial`** puis choisissez Supprimer.

Apprenez à créer une tâche pour les instances gérées Amazon ECS à l'aide du AWS CLI

Les étapes suivantes vous aident à configurer un cluster, créer un fournisseur de capacité, enregistrer une définition de tâche, exécuter une tâche Linux et réaliser d'autres scénarios courants dans Amazon ECS avec des instances gérées Amazon ECS à l'aide de l' AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir de l' AWS CLI et kits SDK AWS , et depuis l'API Amazon ECS à la fois sur IPv4 et IPv6. Pour plus d'informations, consultez the section called "Utilisation des points de terminaison Dual-Stack".

Rubriques

- the section called "Conditions préalables"
- the section called "Étape 1 : créer un cluster"
- the section called "Étape 2 : créer un fournisseur de capacité pour les instances gérées Amazon ECS"
- the section called "Étape 3 : définir une stratégie de fournisseur de capacité par défaut pour le cluster"
- the section called "Étape 4 : enregistrer une définition de tâche Linux"
- the section called "Étape 5 : répertorier les définitions de tâche"
- the section called "Étape 6 : créer un service"
- the section called "Étape 7 : répertorier les services"
- the section called "Étape 8 : décrire le service en cours d'exécution"
- the section called "Étape 9 : tester"

- the section called “Étape 10 : nettoyer”

Conditions préalables

Avant de commencer le didacticiel, vous devez avoir effectué les opérations suivantes :

- Vous avez suivi les étapes décrites dans la section Configuration pour utiliser Amazon ECS.
- La dernière version du AWS CLI est installé et configuré. Pour plus d'informations sur l'installation ou la mise à niveau du AWS CLI, voir Installation ou mise à jour vers la dernière version du AWS CLI dans le AWS Command Line Interface Guide de l'utilisateur.
- Vous devez avoir suivi les étapes de the section called “Configuration”.
- Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :
  - Rôle d'infrastructure : permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

    Pour plus d’informations, consultez Rôle IAM d’infrastructure Amazon ECS.
  - Profil d’instance : fournit des autorisations pour l’agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

    Le nom du rôle d'instance doit inclure `ecsInstanceRole` comme préfixe pour correspondre à `iam:PassRole` action dans le rôle de l'infrastructure.

    Pour plus d’informations, consultez Profil d’instance des instances gérées Amazon ECS.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Amazon ECR Public. Vos instances doivent donc disposer d’un accès Internet. Pour attribuer à vos instances une route vers Internet, utilisez l’une des options suivantes :
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et attribuez une adresse IP publique aux instances.

  Pour plus d’informations, consultez the section called “Créer un Virtual Private Cloud”.

  Pour de plus amples informations sur les règles et les groupes de sécurité, consultez les sections Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l’utilisateur Amazon Virtual Private Cloud.

- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que AWS CloudShell? dans le AWS CloudShell Guide de l'utilisateur.

Étape 1 : créer un cluster

Par défaut, votre compte reçoit un cluster `default`.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name managed-instances-cluster
```

Sortie :

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "managed-instances-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
```

```
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/managed-instances-
cluster"
    }
}
```

Étape 2 : créer un fournisseur de capacité pour les instances gérées Amazon ECS

Avant de pouvoir exécuter des tâches à l'aide d'instances gérées Amazon ECS, vous devez créer un fournisseur de capacité qui définit la configuration de l'infrastructure. Le fournisseur de capacité spécifie les rôles IAM, la configuration réseau et les autres paramètres de vos instances gérées.

Créez un fichier JSON avec la configuration de votre fournisseur de capacité. Remplacez les valeurs des espaces réservés par vos identificateurs de ressources réels :

```
{
    "name": "managed-instances-cp",
    "cluster": "managed-instances-cluster",
    "managedInstancesProvider": {
        "infrastructureRoleArn":
 "arn:aws:iam::aws_account_id:role/ecsInfrastructureRole",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::aws_account_id:instance-
profile/ecsInstanceRole",
            "networkConfiguration": {
                "subnets": [
                    "subnet-abcdef01234567890",
                    "subnet-1234567890abcdef0"
                ],
                "securityGroups": [
                    "sg-0123456789abcdef0"
                ]
            },
            "storageConfiguration": {
                "storageSizeGiB": 100
            },
            "monitoring": "basic"
        }
    }
}
```

Enregistrez cette configuration sous `managed-instances-cp.json` et créez le fournisseur de capacité :

```
aws ecs create-capacity-provider --cli-input-json file://managed-instances-cp.json
```

La commande renvoie une description du fournisseur de capacité une fois sa création terminée.

Étape 3 : définir une stratégie de fournisseur de capacité par défaut pour le cluster

Mettez à jour le cluster pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS comme stratégie de fournisseur de capacité par défaut. Cela permet aux tâches et aux services d'utiliser automatiquement les instances gérées Amazon ECS sans spécifier explicitement le fournisseur de capacité.

Créez un fichier JSON avec la configuration du fournisseur de capacité du cluster :

```
{
    "cluster": "managed-instances-cluster",
    "capacityProviders": [
        "managed-instances-cp"
    ],
    "defaultCapacityProviderStrategy": [
        {
            "capacityProvider": "managed-instances-cp",
            "weight": 1
        }
    ]
}
```

Enregistrez cette configuration sous le nom `cluster-cp-strategy.json` et mettez à jour le cluster :

```
aws ecs put-cluster-capacity-providers --cli-input-json file://cluster-cp-strategy.json
```

Étape 4 : enregistrer une définition de tâche Linux

Avant de pouvoir exécuter une tâche sur votre cluster, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui crée une application Web PHP à l'aide de l'image de conteneur

httpd hébergée sur Docker Hub. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez the section called “Paramètres de définition des tâches pour Fargate”.

```
{
    "family": "sample-managed-instances",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "managed-instances-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "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 using Amazon ECS
 Managed Instances.</p></div></body></html>' >  /usr/local/apache2/htdocs/index.html &&
 httpd-foreground\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "MANAGED_INSTANCES"
    ],
    "cpu": "256",
    "memory": "512"
}
```

Enregistrez la définition de tâche JSON sous forme de fichier et transmettez-la avec l'option `--cli-input-json file://`*`path_to_file.json`*.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/managed-instances-task.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 5 : répertorier les définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez run-task ou start-task.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-managed-instances:1"
    ]
}
```

Étape 6 : créer un service

Une fois que vous avez enregistré une tâche pour votre compte, vous pouvez créer un service pour la tâche enregistrée dans votre cluster. Pour cet exemple, vous créez un service avec une instance de la définition de tâche `sample-managed-instances:1` exécutée dans votre cluster. La tâche nécessite un itinéraire vers Internet. Il y a deux façons d'y parvenir. Une façon consiste à utiliser un sous-réseau privé configuré avec une passerelle NAT avec une adresse IP Elastic dans un sous-réseau public. Une autre façon consiste à utiliser un sous-réseau public et à attribuer une adresse IP publique à votre tâche. Les deux exemples sont présentés ci-dessous.

Exemple d'utilisation d'un sous-réseau privé :

```
aws ecs create-service --cluster managed-instances-cluster --service-name managed-instances-service --task-definition sample-managed-instances:1 --desired-count 1 --network-configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-abcd1234]}"
```

Exemple d'utilisation d'un sous-réseau public :

```
aws ecs create-service --cluster managed-instances-cluster --service-name managed-
instances-service --task-definition sample-managed-instances:1 --desired-
count 1 --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

La commande create-service renvoie une description du service une fois sa création terminée.

Étape 7 : répertorier les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter le nom du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster managed-instances-cluster
```

Sortie :

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/managed-instances-cluster/managed-
instances-service"
    ]
}
```

Étape 8 : décrire le service en cours d'exécution

Décrivez le service à l'aide du nom de service récupéré précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-services --cluster managed-instances-cluster --services managed-
instances-service
```

En cas de réussite, une description des défaillances de service et des services est renvoyée. Par exemple, dans la section `services`, vous trouverez des informations sur les déploiements, telles que le statut des tâches en cours d'exécution ou en attente. Vous trouverez également des informations sur la définition des tâches, la configuration réseau et les événements horodatés. Dans

la section des défaillances, vous trouverez des informations sur les défaillances, le cas échéant, associées à l'appel.

Le résultat indiquera que le service utilise le fournisseur de capacité d'instances gérées Amazon ECS :

```
{
    "services": [
        {
            "capacityProviderStrategy": [
                {
                    "capacityProvider": "managed-instances-cp",
                    "weight": 1,
                    "base": 0
                }
            ],
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "ENABLED"
                }
            },
            "enableECSManagedTags": false,
            "loadBalancers": [],
            "deploymentController": {
                "type": "ECS"
            },
            "desiredCount": 1,
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/managed-instances-
cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/managed-instances-
cluster/managed-instances-service",
            "serviceName": "managed-instances-service",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-managed-instances:1"
        }
    ],
    "failures": []
```

```
}
```

Étape 9 : tester

Pour tester votre déploiement, vous devez trouver l'adresse IP publique de l'instance gérée qui exécute votre tâche.

Tâche de test déployée à l'aide d'un sous-réseau public

Tout d'abord, obtenez l'ARN de la tâche auprès de votre service :

```
aws ecs list-tasks --cluster managed-instances-cluster --service managed-instances-
service
```

La sortie contient l'ARN de la tâche :

```
{
    "taskArns": [
        "arn:aws:ecs:region:aws_account_id:task/managed-instances-cluster/EXAMPLE"
    ]
}
```

Décrivez la tâche pour obtenir l'ARN de l'instance de conteneur. Utilisez l'ARN de la tâche pour le paramètre `tasks` :

```
aws ecs describe-tasks --cluster managed-instances-cluster --tasks
 arn:aws:ecs:region:aws_account_id:task/managed-instances-cluster/EXAMPLE
```

Le résultat indique que la tâche est exécutée sur des instances gérées Amazon ECS et inclut l'ARN de l'instance de conteneur :

```
{
    "tasks": [
        {
            "launchType": "MANAGED_INSTANCES",
            "capacityProviderName": "managed-instances-cp",
            "containerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/managed-instances-cluster/CONTAINER_INSTANCE_ID",
            "taskArn": "arn:aws:ecs:region:aws_account_id:task/managed-instances-
cluster/EXAMPLE",
```

```
            "taskDefinitionArn": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-managed-instances:1"
        }
    ]
}
```

Décrivez l'instance de conteneur pour obtenir l'ID d'instance EC2 :

```
aws ecs describe-container-instances --cluster managed-instances-cluster --container-
instances CONTAINER_INSTANCE_ID
```

La sortie comprend l'ID d'instance EC2 :

```
{
    "containerInstances": [
        {
            "ec2InstanceId": "i-1234567890abcdef0",
            "capacityProviderName": "managed-instances-cp",
            "containerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/managed-instances-cluster/CONTAINER_INSTANCE_ID"
        }
    ]
}
```

Décrivez l'instance EC2 pour obtenir l'adresse IP publique :

```
aws ec2 describe-instances --instance-ids i-1234567890abcdef0
```

L'adresse IP publique se trouve dans la sortie :

```
{
    "Reservations": [
        {
            "Instances": [
                {
                    "PublicIpAddress": "198.51.100.2",
                    "InstanceId": "i-1234567890abcdef0"
                }
            ]
        }
    ]
```

```
}
```

Entrez l'adresse IP publique dans votre navigateur Web et vous devriez voir une page Web qui affiche l'application Amazon ECS exemple s'exécutant sur les instances gérées Amazon ECS.

Tâche de test déployée à l'aide d'un sous-réseau privé

Pour les tâches déployées dans des sous-réseaux privés, vous pouvez utiliser Amazon ECS Exec pour vous connecter au conteneur et tester le déploiement depuis l'instance. Suivez les mêmes étapes précédentes pour obtenir l'ARN de la tâche, puis utilisez ECS Exec :

```
aws ecs execute-command --cluster managed-instances-cluster \
    --task arn:aws:ecs:region:aws_account_id:task/managed-instances-cluster/EXAMPLE \
    --container managed-instances-app \
    --interactive \
    --command "/bin/sh"
```

Une fois le shell interactif lancé, vous pouvez tester le serveur Web :

```
curl localhost
```

Vous devriez voir l'équivalent HTML de la page Web d'exemple d'application Amazon ECS.

Étape 10 : nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Supprimez le service de la fonction :

```
aws ecs delete-service --cluster managed-instances-cluster --service managed-instances-
service --force
```

Attendez que le service soit supprimé et que toutes les tâches s'arrêtent, puis supprimez le fournisseur de capacité :

```
aws ecs delete-capacity-provider --capacity-provider managed-instances-cp
```

Supprimez le cluster :

```
aws ecs delete-cluster --cluster managed-instances-cluster
```

> **Note**
>
> Les instances gérées sont automatiquement résiliées lorsque le fournisseur de capacité est supprimé. Vous n'avez pas besoin de résilier manuellement les instances EC2.

Création d'une tâche Amazon ECS Linux pour Fargate

Amazon Elastic Container Service (Amazon ECS) est un service de gestion de conteneurs hautement évolutif et rapide, qui facilite l'exécution, l'arrêt et la gestion de vos conteneurs. Vous pouvez héberger vos conteneurs sur une infrastructure sans serveur gérée par Amazon ECS en lançant vos services ou vos tâches sur AWS Fargate. Pour plus d'informations sur Fargate, consultez la section Architecte pour AWS Fargate pour Amazon ECS.

Commencez à utiliser Amazon ECS sur AWS Fargate en utilisant Fargate pour vos tâches dans les régions où Amazon ECS prend en charge AWS Fargate.

Procédez comme suit pour démarrer avec Amazon ECS sur AWS Fargate.

Conditions préalables

Avant de commencer, assurez-vous d'avoir effectué les étapes décrites dans Configurer l'utilisation d'Amazon ECS et que votre utilisateur IAM dispose des autorisations spécifiées dans l'exemple de politique IAM `AdministratorAccess`.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour plus d'informations, consultez Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS Service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Création d'un groupe de sécurité pour votre instance Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.
>
> - Type : HTTP
> - Protocole : TCP
> - Plage de ports : 80
> - Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer le cluster

Créez un cluster qui utilise le VPC par défaut.

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour plus d'informations, consultez the section called "Exemples de cluster Amazon ECS".

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez un nom unique.

   Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
6. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
7. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

- Pour Key (Clé), saisissez le nom de la clé.
- Pour Value (Valeur), saisissez la valeur de clé.

[Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.

8. Cliquez sur Créer.

Étape 2 : Créer une définition de tâche

Une définition de tâche est similaire au modèle d'une application. Chaque fois que vous lancez une tâche dans Amazon ECS, vous spécifiez une définition de tâche. Le service connaît ensuite l'image Docker à utiliser pour les conteneurs, le nombre de conteneurs à utiliser dans la tâche et l'allocation des ressources pour chaque conteneur.

1. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
2. Choisissez Create new Task Definition (Créer une nouvelle définition de tâche), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).
3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "fargate-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "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\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

4. Cliquez sur Créer.

Étape 3 : Créer le service

Créez un service à l'aide de la définition de tâche.

1. Dans le volet de navigation, choisissez Clusters, puis sélectionnez le cluster que vous avez créé dans Étape 1 : créer le cluster.
2. Sous l'onglet Services choisissez Create (Créer).
3. En dessous Détails du service, spécifiez la manière dont votre application est déployée.
   a. Pour Famille de définitions de tâches, choisissez la définition de tâche que vous avez créée dans Étape 2 : Créer une définition de tâche.
   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
4. En dessous Environnement, choisissez Type de lancement puis choisissez `FARGATE`.
5. En dessous Configuration du déploiement, pour Tâches souhaitées, entrez 1.
6. Sous Mise en réseau, vous pouvez créer un nouveau groupe de sécurité ou sélectionner un groupe de sécurité existant pour votre tâche. Assurez-vous que le groupe de sécurité que vous utilisez possède la règle entrante répertoriée sous Conditions préalables.
7. Cliquez sur Créer.

Étape 4 : Afficher votre service

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.
4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer le service.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte.

Étape 5 : nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant le CloudFormation pile qui les a créés.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer). Vous pouvez également utiliser l'option `Force delete` pour qu'Amazon ECS réduise le service en votre nom avant de le supprimer.

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, entrez supprimer ***`cluster-name`***, puis choisissez Supprimer. La suppression du cluster nettoie les ressources associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Création d’une tâche Windows Amazon ECS pour Fargate

Commencez à utiliser Amazon ECS sur AWS Fargate en utilisant Fargate pour vos tâches dans les régions où Amazon ECS prend en charge AWS Fargate.

Procédez comme suit pour démarrer avec Amazon ECS sur AWS Fargate.

Conditions préalables

Avant de commencer, assurez-vous d’avoir effectué les étapes décrites dans Configurer l'utilisation d'Amazon ECS et que votre utilisateur IAM dispose des autorisations spécifiées dans l’exemple de politique IAM `AdministratorAccess`.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour plus d’informations, consultez Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS Service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d’informations sur la création d’un groupe de sécurité, consultez la section Création d’un groupe de sécurité pour votre instance Amazon EC2 dans le Guide de l’utilisateur Amazon EC2.
>
> - Type : HTTP
> - Protocole : TCP
> - Plage de ports : 80
> - Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer un cluster

Vous pouvez créer un cluster appelé windows qui utilise le VPC par défaut.

Pour créer un cluster avec Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez windows.
6. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
7. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
8. Cliquez sur Créer.

Étape 2 : Enregistrer une définition de tâche Windows

Avant de pouvoir exécuter des conteneurs Windows dans votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. L'exemple de définition de tâche suivant affiche une page web simple sur le port 8080 d'une instance de conteneur avec l'image de conteneur `mcr.microsoft.com/windows/servercore/iis`.

Pour enregistrer l'exemple de définition de tâche auprès du Console de gestion AWS

1. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).

2. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<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>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

4. Vérifiez vos informations et choisissez Create (Créer).

Étape 3 : Créer un service avec votre définition de tâche

Après avoir enregistré votre définition de tâche, vous pouvez placer des tâches dans votre cluster avec elle. La procédure suivante crée un service avec votre définition de tâche et place une tâche dans votre cluster.

Pour créer un service à partir de votre définition de tâche avec la console

1. Dans le volet de navigation, choisissez Clusters, puis sélectionnez le cluster que vous avez créé dans Étape 1 : créer un cluster.
2. Sous l'onglet Services choisissez Create (Créer).
3. Sous Deployment configuration (Configuration du déploiement), spécifiez la manière dont votre application est déployée.
    a. Pour Task Definition (Définition de tâche), choisissez la définition de tâche que vous avez créé dans Étape 2 : Enregistrer une définition de tâche Windows.
    b. Pour Service name (Nom du service), saisissez un nom pour votre service.
    c. Pour Desired tasks (Tâches souhaitées), saisissez 1.
4. Sous Mise en réseau, vous pouvez créer un groupe de sécurité ou sélectionner un groupe de sécurité existant. Assurez-vous que le groupe de sécurité que vous utilisez possède la règle entrante répertoriée sous Conditions préalables.
5. Cliquez sur Créer.

Étape 4 : Afficher votre service

Lorsque votre service a lancé une tâche dans votre cluster, vous pouvez afficher le service et ouvrir la page de test IIS dans un navigateur pour vérifier que le conteneur est en cours d'exécution.

> **Note**
>
> Cette opération peut prendre jusqu'à 15 minutes pour que votre instance de conteneur télécharge et extrait les couches de base du conteneur Windows.

Pour afficher votre service

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.
4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer un service avec votre définition de tâche.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte.

Étape 5 : Nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant le CloudFormation pile qui les a créés.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer).

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, entrez supprimer ***`cluster-name`***, puis choisissez Supprimer. La suppression du cluster nettoie les ressources associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Création d’une tâche Windows Amazon ECS pour EC2

Commencez à utiliser Amazon ECS avec EC2 en enregistrant une définition de tâche, en créant un cluster et en créant un service dans la console.

Procédez comme suit pour démarrer avec Amazon ECS en utilisant le type de lancement EC2.

Conditions préalables

Avant de commencer, assurez-vous d'avoir effectué les étapes décrites dans Configurer l'utilisation d'Amazon ECS et que votre utilisateur IAM dispose des autorisations spécifiées dans l'exemple de politique IAM `AdministratorAccess`.

La console tente de créer automatiquement le rôle IAM d'exécution de tâche, qui est requis pour les tâches Fargate. Pour garantir que la console peut créer ce rôle IAM, l'une des conditions suivantes doit être remplie :

- Votre utilisateur dispose d'un accès administrateur. Pour plus d'informations, consultez Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer une fonction de service. Pour plus d'informations, voir Création d'un rôle pour déléguer des autorisations à un AWS Service.
- Un utilisateur disposant d'un accès administrateur a créé manuellement le rôle d'exécution de tâche afin qu'il soit disponible sur le compte à utiliser. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Important

Le groupe de sécurité que vous sélectionnez lors de la création d'un service avec votre définition de tâche doit avoir le port 80 ouvert pour le trafic entrant. Ajoutez les règles entrantes suivantes à votre groupe de sécurité. Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Création d'un groupe de sécurité pour votre instance Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

- Type : HTTP
- Protocole : TCP
- Plage de ports : 80
- Source : n'importe où (`0.0.0.0/0`)

Étape 1 : créer un cluster

Un cluster Amazon ECS est un regroupement logique de tâches, de services et d'instances de conteneur.

Les étapes suivantes vous expliquent comment créer un cluster avec une instance Amazon EC2 enregistrée dans celui-ci, ce qui nous permettra d'exécuter une tâche dessus. Si un champ spécifique n'est pas mentionné, conservez les valeurs par défaut utilisées par la console.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour plus d’informations, consultez the section called “Exemples de cluster Amazon ECS”.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez un nom unique.

   Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
6. (Facultatif) Pour modifier le VPC et les sous-réseaux d'où vos tâches et services se lancent, sous Networking (Réseaux), effectuez l'une des opérations suivantes :

   - Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
   - Pour passer à un VPC autre celui par défaut, sous VPC, choisissez un VPC existant, puis sous Subnets (Sous-réseaux), sélectionnez chaque sous-réseau.
7. Pour ajouter des instances Amazon EC2 à votre cluster, développez Infrastructure, puis sélectionnez Instances Amazon EC2. Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :

   a. Pour utiliser un groupe Auto Scaling existant, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez le groupe.

b. Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :

- Pour Fonctionnement system/Architecture, choisissez Amazon ECS-optimized AMI pour les instances du groupe Auto Scaling.
- Pour EC2 instance type (Type d'instance EC2), choisissez le type d'instance pour vos charges de travail. Pour plus d'informations sur les différents types d'instances et leurs cas d'utilisation, veuillez consulter Instances Amazon EC2.

  La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.
- Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
- Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling. Les instances Amazon EC2 entraînent des coûts tant qu'elles existent dans votre AWS ressources. Pour plus d'informations, consultez Tarification Amazon EC2.

8. (Facultatif) Pour activer Container Insights, développez Monitoring (Surveillance), puis activez Use Container Insights (Utiliser Container Insights).
9. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

   [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
10. Cliquez sur Créer.

Étape 2 : Enregistrer une définition de tâche

Pour enregistrer l'exemple de définition de tâche auprès du Console de gestion AWS

1. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).

2. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
3. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Enregistrer.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html
 -Type file -Value '<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>'; C:\\ServiceMonitor.exe w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
}
```

4. Vérifiez vos informations et choisissez Create (Créer).

Étape 3 : Créer un service

Un service Amazon ECS service vous aide à exécuter et à gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de vos tâches est défaillante ou s'arrête pour une raison quelconque, le planificateur Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer et maintenir le nombre souhaité de tâches dans le service. Pour de plus amples informations sur les services, consultez Services Amazon ECS.

Pour créer un service

1. Dans le panneau de navigation, choisissez Clusters.
2. Sélectionnez le cluster que vous avez créé dans Étape 1 : créer un cluster.
3. Dans l'onglet Services choisissez Create (Créer).
4. Dans la section Environment (Environnement), procédez comme suit :
    a. Pour les Compute options (Options de calcul), choisissez Launch type (Type de lancement).
    b. Pour Launch type (Type de lancement), sélectionnez EC2
5. Dans la section Deployment configuration (Configuration de déploiement), effectuez les opérations suivantes :
    a. Pour Family (Famille), choisissez la définition de tâche que vous avez créé dans Étape 2 : Enregistrer une définition de tâche.
    b. Pour Service name (Nom du service), saisissez un nom pour votre service.
    c. Pour Desired tasks (Tâches souhaitées), saisissez 1.
6. Vérifiez les options et choisissez Create (Créer).
7. Choisissez View service (Afficher le service) pour vérifier votre service.

Étape 4 : Afficher votre service

Le service est une application web qui vous permet d'afficher ses conteneurs à l'aide d'un navigateur web.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Choisissez le cluster sur lequel vous avez exécuté le service.

4. Dans l'onglet Services, sous Nom du service, choisissez le service que vous avez créé dans Étape 3 : Créer un service.
5. Cliquez sur l'onglet Tâches, puis sélectionnez la tâche dans votre service.
6. Sur la page des tâches, dans la section Configuration, sous IP publique, choisissez Adresse ouverte. La capture d'écran ci-dessous représente la sortie attendue.



Étape 5 : Nettoyer

Lorsque vous avez fini d'utiliser un cluster Amazon ECS, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Certaines ressources Amazon ECS, comme les tâches, les services, les clusters et les instances de conteneur, sont nettoyées à l'aide de la console Amazon ECS. Les autres ressources, telles que les instances Amazon EC2, les équilibreurs de charge Elastic Load Balancing et les groupes Auto Scaling, doivent être nettoyées manuellement dans la console Amazon EC2 ou en supprimant le CloudFormation pile qui les a créés.

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, sélectionnez le cluster que vous avez créé pour ce didacticiel.
3. Choisissez l'onglet Services.
4. Sélectionnez le service, puis choisissez Supprimer.
5. À l'invite de confirmation, saisissez delete (supprimer) puis choisissez Delete (Supprimer).

   Attendez que le service soit supprimé.
6. Choisissez Delete Cluster (Supprimer le cluster). À l'invite de confirmation, entrez supprimer ***cluster-name***, puis choisissez Supprimer. La suppression du cluster nettoie les ressources

associées qui ont été créées avec le cluster, y compris les groupes Auto Scaling, les VPC ou les équilibreurs de charge.

Création de ressources Amazon ECS à l'aide du AWS CDK

Le Cloud Development Kit AWS (AWS CDK) est un Infrastructure-as-Code cadre (IAC) que vous pouvez utiliser pour définir AWS infrastructure cloud en utilisant le langage de programmation de votre choix. Pour définir votre propre infrastructure de cloud, écrivez d'abord une application (dans l'un des langages pris en charge par le CDK) contenant une ou plusieurs piles. Ensuite, vous le synthétisez en un CloudFormation modélisez et déployez vos ressources sur votre Compte AWS. Suivez les étapes décrites dans cette rubrique pour déployer un serveur Web conteneurisé avec Amazon Elastic Container Service (Amazon ECS) et AWS CDK sur Fargate.

Le AWS La bibliothèque Construct, incluse dans le CDK, fournit des modules que vous pouvez utiliser pour modéliser les ressources qui Services AWS fournir. Pour les services populaires, la bibliothèque fournit des constructions organisées avec des valeurs par défaut intelligentes et les bonnes pratiques. L'un de ces modules, en particulier `aws-ecs-patterns`, fournit des abstractions de haut niveau que vous pouvez utiliser pour définir votre service conteneurisé et toutes les ressources de support nécessaires en quelques lignes de code.

Cette rubrique utilise la construction `ApplicationLoadBalancedFargateService`. Cette construction déploie un Amazon ECS service sur Fargate derrière un Application Load Balancer. Le module `aws-ecs-patterns` inclut également des constructions qui utilisent un Network Load Balancer et s'exécutent sur Amazon EC2.

Avant de commencer cette tâche, configurez votre AWS CDK environnement de développement et installez le AWS CDK en exécutant la commande suivante. Pour obtenir des instructions sur la configuration de votre AWS CDK environnement de développement, voir Commencer à utiliser le AWS CDK - Prérequis.

```
npm install -g aws-cdk
```

> **Note**
>
> Ces instructions supposent que vous utilisez AWS CDK v2.

Rubriques

- Étape 1 : configurez votre AWS CDK project
- Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate
- Étape 3 : Tester le service web
- Étape 4 : nettoyer
- Étapes suivantes

Étape 1 : configurez votre AWS CDK project

Créez un répertoire pour votre nouveau AWS CDK application et initialisez le projet.

TypeScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language typescript
```

JavaScript

```
mkdir hello-ecs
cd hello-ecs
cdk init --language javascript
```

Python

```
mkdir hello-ecs
cd hello-ecs
cdk init --language python
```

Une fois le projet démarré, activez l'environnement virtuel du projet et installez AWS CDK les dépendances de base de l'entreprise.

```
source .venv/bin/activate
python -m pip install -r requirements.txt
```

Java

```
mkdir hello-ecs
cd hello-ecs
```

```
cdk init --language java
```

Importez ce projet Maven dans votre IDE Java. Par exemple, dans Eclipse, utilisez Fichier >Importer >Maven >Projets Maven existants.

C#

```
mkdir hello-ecs
cd hello-ecs
cdk init --language csharp
```

Go

```
mkdir hello-ecs
cd hello-ecs
cdk init --language go
```

> **Note**
>
> Le AWS CDK Un modèle d'application utilise le nom du répertoire du projet pour générer des noms pour les fichiers source et les classes. Dans cet exemple, le répertoire est nommé `hello-ecs`. Si vous utilisez un nom répertoire différent, votre application ne correspondra pas à ces instructions.

AWS CDK La v2 inclut des constructions stables pour tous Services AWS dans un seul package appelé `aws-cdk-lib`. Ce package est installé en tant que dépendance lorsque vous initialisez le projet. Lorsque vous travaillez avec certains langages de programmation, le package est installé lorsque vous créez le projet pour la première fois. Cette rubrique explique comment utiliser une construction Amazon ECS Patterns, qui fournit des abstractions de haut niveau pour travailler avec Amazon ECS. Ce module s'appuie sur les constructions Amazon ECS et d'autres constructions pour allouer les ressources nécessaires à votre application Amazon ECS.

Les noms que vous utilisez pour importer ces bibliothèques dans votre application CDK peuvent différer légèrement selon le langage de programmation que vous utilisez. À titre de référence, les noms suivants sont utilisés dans chaque langage de programmation CDK pris en charge.

TypeScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

JavaScript

```
aws-cdk-lib/aws-ecs
aws-cdk-lib/aws-ecs-patterns
```

Python

```
aws_cdk.aws_ecs
aws_cdk.aws_ecs_patterns
```

Java

```
software.amazon.awscdk.services.ecs
software.amazon.awscdk.services.ecs.patterns
```

C#

```
Amazon.CDK.AWS.ECS
Amazon.CDK.AWS.ECS.Patterns
```

Go

```
github.com/aws/aws-cdk-go/awscdk/v2/awsecs
github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns
```

Étape 2 : utilisez le AWS CDK pour définir un serveur Web conteneurisé sur Fargate

Utilisez l'image du conteneur `amazon-ecs-sample`. Cette image contient une application Web PHP qui s'exécute sur nginx.

Dans le AWS CDK projet que vous avez créé, modifiez le fichier contenant la définition de la pile pour qu'il ressemble à l'un des exemples suivants.

> **Note**
>
> Une pile est une unité de déploiement. Toutes les ressources doivent se trouver dans une pile et toutes les ressources d'une pile sont déployées en même temps. Si une ressource ne parvient pas à se déployer, toutes les autres ressources déjà déployées sont annulées. Un AWS CDK l'application peut contenir plusieurs piles, et les ressources d'une pile peuvent faire référence à des ressources d'une autre pile.

TypeScript

Mise à jour `lib/hello-ecs-stack.ts` de sorte qu'il ressemble à ce qui suit.

```
import * as cdk from 'aws-cdk-lib';
import { Construct } from 'constructs';
import * as ecs from 'aws-cdk-lib/aws-ecs';
import * as ecsp from 'aws-cdk-lib/aws-ecs-patterns';

export class HelloEcsStack extends cdk.Stack {
  constructor(scope: Construct, id: string, props?: cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('public.ecr.aws/ecs-sample-image/
amazon-ecs-sample:latest'),
      },
      publicLoadBalancer: true
    });
  }
}
```

JavaScript

Mise à jour `lib/hello-ecs-stack.js` de sorte qu'il ressemble à ce qui suit.

```
const cdk = require('aws-cdk-lib');
const { Construct } = require('constructs');
const ecs = require('aws-cdk-lib/aws-ecs');
const ecsp = require('aws-cdk-lib/aws-ecs-patterns');

class HelloEcsStack extends cdk.Stack {
```

```
  constructor(scope = Construct, id = string, props = cdk.StackProps) {
    super(scope, id, props);

    new ecsp.ApplicationLoadBalancedFargateService(this, 'MyWebServer', {
      taskImageOptions: {
        image: ecs.ContainerImage.fromRegistry('amazon/amazon-ecs-sample'),
      },
      publicLoadBalancer: true
    });
  }
}

module.exports = { HelloEcsStack }
```

Python

Mise à jour `hello-ecs/hello_ecs_stack.py` de sorte qu'il ressemble à ce qui suit.

```
import aws_cdk as cdk
from constructs import Construct

import aws_cdk.aws_ecs as ecs
import aws_cdk.aws_ecs_patterns as ecsp

class HelloEcsStack(cdk.Stack):

    def __init__(self, scope: Construct, construct_id: str, **kwargs) -> None:
        super().__init__(scope, construct_id, **kwargs)

        ecsp.ApplicationLoadBalancedFargateService(self, "MyWebServer",
            task_image_options=ecsp.ApplicationLoadBalancedTaskImageOptions(
                image=ecs.ContainerImage.from_registry("amazon/amazon-ecs-sample")),
            public_load_balancer=True
        )
```

Java

Mise à jour `src/main/java/com.myorg/HelloEcsStack.java` de sorte qu'il ressemble à ce qui suit.

```
package com.myorg;

import software.constructs.Construct;
```

```
import software.amazon.awscdk.Stack;
import software.amazon.awscdk.StackProps;

import software.amazon.awscdk.services.ecs.ContainerImage;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedFargateService;
import
 software.amazon.awscdk.services.ecs.patterns.ApplicationLoadBalancedTaskImageOptions;

public class HelloEcsStack extends Stack {
    public HelloEcsStack(final Construct scope, final String id) {
        this(scope, id, null);
    }

    public HelloEcsStack(final Construct scope, final String id, final StackProps
 props) {
        super(scope, id, props);

        ApplicationLoadBalancedFargateService.Builder.create(this, "MyWebServer")
         .taskImageOptions(ApplicationLoadBalancedTaskImageOptions.builder()
           .image(ContainerImage.fromRegistry("amazon/amazon-ecs-sample"))
           .build())
         .publicLoadBalancer(true)
         .build();
    }
}
```

C#

Mise à jour `src/HelloEcs/HelloEcsStack.cs` de sorte qu'il ressemble à ce qui suit.

```
using Amazon.CDK;
using Constructs;
using Amazon.CDK.AWS.ECS;
using Amazon.CDK.AWS.ECS.Patterns;
namespace HelloEcs
{
    public class HelloEcsStack : Stack
    {
        internal HelloEcsStack(Construct scope, string id, IStackProps props =
 null) : base(scope, id, props)
        {
            new ApplicationLoadBalancedFargateService(this, "MyWebServer",
                new ApplicationLoadBalancedFargateServiceProps
```

```
                {
                    TaskImageOptions = new ApplicationLoadBalancedTaskImageOptions
                    {
                        Image = ContainerImage.FromRegistry("amazon/amazon-ecs-
sample")
                    },
                    PublicLoadBalancer = true
                });
        }
    }
}
```

Go

Mise à jour `hello-ecs.go` de sorte qu'il ressemble à ce qui suit.

```
package main

import (
 "github.com/aws/aws-cdk-go/awscdk/v2"
 // "github.com/aws/aws-cdk-go/awscdk/v2/awssqs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecs"
 "github.com/aws/aws-cdk-go/awscdk/v2/awsecspatterns"
 "github.com/aws/constructs-go/constructs/v10"
 "github.com/aws/jsii-runtime-go"
)

type HelloEcsStackProps struct {
 awscdk.StackProps
}

func NewHelloEcsStack(scope constructs.Construct, id string, props
 *HelloEcsStackProps) awscdk.Stack {
 var sprops awscdk.StackProps
 if props != nil {
  sprops = props.StackProps
 }
 stack := awscdk.NewStack(scope, &id, &sprops)

 // The code that defines your stack goes here

 // example resource
 // queue := awssqs.NewQueue(stack, jsii.String("HelloEcsQueue"),
 &awssqs.QueueProps{
```

```
 //  VisibilityTimeout: awscdk.Duration_Seconds(jsii.Number(300)),
 // })
 res := awsecspatterns.NewApplicationLoadBalancedFargateService(stack,
 jsii.String("MyWebServer"),
  &awsecspatterns.ApplicationLoadBalancedFargateServiceProps{
   TaskImageOptions: &awsecspatterns.ApplicationLoadBalancedTaskImageOptions{
    Image: awsecs.ContainerImage_FromRegistry(jsii.String("amazon/amazon-ecs-
sample"), &awsecs.RepositoryImageProps{}),
   },
  },
 )
 awscdk.NewCfnOutput(stack, jsii.String("LoadBalancerDNS"),
 &awscdk.CfnOutputProps{Value: res.LoadBalancer().LoadBalancerDnsName()})

 return stack
}

func main() {
 defer jsii.Close()

 app := awscdk.NewApp(nil)

 NewHelloEcsStack(app, "HelloEcsStack", &HelloEcsStackProps{
  awscdk.StackProps{
   Env: env(),
  },
 })

 app.Synth(nil)
}

// env determines the AWS environment (account+region) in which our stack is to
// be deployed. For more information see: https://docs.aws.amazon.com/cdk/latest/
guide/environments.html
func env() *awscdk.Environment {
 // If unspecified, this stack will be "environment-agnostic".
 // Account/Region-dependent features and context lookups will not work, but a
 // single synthesized template can be deployed anywhere.
 //---------------------------------------------------------------------------
 return nil

 // Uncomment if you know exactly what account and region you want to deploy
 // the stack to. This is the recommendation for production stacks.
 //---------------------------------------------------------------------------
```

```
 // return &awscdk.Environment{
 //  Account: jsii.String("123456789012"),
 //  Region:  jsii.String("us-east-1"),
 // }

 // Uncomment to specialize this stack for the AWS Account and Region that are
 // implied by the current CLI configuration. This is recommended for dev
 // stacks.
 //---------------------------------------------------------------------------
 // return &awscdk.Environment{
 //  Account: jsii.String(os.Getenv("CDK_DEFAULT_ACCOUNT")),
 //  Region:  jsii.String(os.Getenv("CDK_DEFAULT_REGION")),
 // }
}
```

Le court extrait précédent inclut les éléments suivants :

- Le nom logique du service : `MyWebServer`.
- L'image du conteneur, obtenue à partir de la galerie publique Amazon ECR :`amazon/amazon-ecs-sample`.
- D'autres informations pertinentes, telles que le fait que l'équilibreur de charge a une adresse publique et est accessible à partir d'Internet.

Le AWS CDK créera toutes les ressources nécessaires au déploiement du serveur Web, y compris les ressources suivantes. Ces ressources ont été omises dans cet exemple.

- Cluster Amazon ECS
- Instances Amazon VPC et Amazon EC2
- Groupe Auto Scaling
- Application Load Balancer
- Rôles et politiques IAM

Certaines ressources allouées automatiquement sont partagées par tous les Amazon ECS services définis dans la pile.

Enregistrez le fichier source, puis exécutez la commande `cdk synth` dans le répertoire principal de l'application. Le AWS CDK lance l'application et synthétise un CloudFormation modèle à partir

de celui-ci, puis affiche le modèle. Le modèle est un fichier YAML d'environ 600 lignes. Le début du fichier est illustré ici. Votre modèle peut être différent de cet exemple.

```
Resources:
  MyWebServerLB3B5FD3AB:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      LoadBalancerAttributes:
        - Key: deletion_protection.enabled
          Value: "false"
      Scheme: internet-facing
      SecurityGroups:
        - Fn::GetAtt:
            - MyWebServerLBSecurityGroup01B285AA
            - GroupId
      Subnets:
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1Subnet3C273B99
        - Ref: EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2Subnet95FF715A
      Type: application
    DependsOn:
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet1DefaultRouteFF4E2178
      - EcsDefaultClusterMnL3mNNYNVpcPublicSubnet2DefaultRouteB1375520
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/Resource
  MyWebServerLBSecurityGroup01B285AA:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Automatically created Security Group for ELB
 HelloEcsStackMyWebServerLB06757F57
      SecurityGroupIngress:
        - CidrIp: 0.0.0.0/0
          Description: Allow from anyone on port 80
          FromPort: 80
          IpProtocol: tcp
          ToPort: 80
      VpcId:
        Ref: EcsDefaultClusterMnL3mNNYNVpc7788A521
    Metadata:
      aws:cdk:path: HelloEcsStack/MyWebServer/LB/SecurityGroup/Resource
# and so on for another few hundred lines
```

Pour déployer le service dans votre Compte AWS, exécutez le `cdk deploy` commande dans le répertoire principal de votre application. Vous êtes invité à approuver les politiques IAM que le AWS CDK généré.

Le déploiement dure plusieurs minutes, au cours desquelles AWS CDK crée plusieurs ressources. Les dernières lignes de la sortie du déploiement incluent le nom d'hôte public de l'équilibreur de charge et l'URL de votre nouveau serveur web. Ce sont les suivants.

```
Outputs:
HelloEcsStack.MyWebServerLoadBalancerDNSXXXXXXX = Hello-MyWeb-ZZZZZZZZZZZZ-
ZZZZZZZZZ.us-west-2.elb.amazonaws.com
HelloEcsStack.MyWebServerServiceURLYYYYYYYY = http://Hello-MyWeb-ZZZZZZZZZZZZ-
ZZZZZZZZZ.us-west-2.elb.amazonaws.com
```

Étape 3 : Tester le service web

Copiez l'URL de la sortie de déploiement et collez-la dans votre navigateur web. Le message de bienvenue suivant provenant du serveur web s'affiche.



Étape 4 : nettoyer

Une fois que vous avez terminé d'utiliser le serveur web, arrêtez le service à l'aide du CDK en exécutant la commande `cdk destroy` dans le répertoire principal de votre application. Cela vous évite de subir des frais imprévus à l'future.

Étapes suivantes

Pour en savoir plus sur la façon de développer AWS infrastructure utilisant le AWS CDK, consultez le AWS CDK Guide du développeur.

Pour plus d'informations sur l'écriture AWS CDK applications dans la langue de votre choix, consultez les rubriques suivantes :

TypeScript

- En collaboration avec le AWS CDK dans TypeScript

JavaScript

- En collaboration avec le AWS CDK dans JavaScript

Python

- En collaboration avec le AWS CDK en Python

Java

- En collaboration avec le AWS CDK en Java

C#

- En collaboration avec le AWS CDK en C#

Go

- En collaboration avec le AWS CDK en Go

Pour plus d'informations sur le AWS Modules Construct Library utilisés dans cette rubrique, voir ce qui suit AWS CDK Aperçus des références d'API.

- aws-ecs
- aws-ecs-patterns

Création de ressources Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot

AWS Interface de ligne de commande Copilot End-of-Support Remarque

AWS La CLI Copilot atteindra la fin du support le 12 juin 2026. Après cette date, l'outil ne bénéficiera plus de mises à jour, de correctifs de sécurité ou de support technique.

- Aucune nouvelle fonctionnalité ou amélioration après le 12 juin 2026
- Aucune mise à jour de sécurité ni correction de bogues
- Pas de support technique
- Les déploiements existants continueront de fonctionner, mais sans maintenance continue

Vous avez besoin d'aide ? Contacter AWS Soutenez ou visitez notre billet de blog pour obtenir des conseils de migration détaillés.

Le AWS Les commandes de l'interface de ligne de commande (CLI) Copilot simplifient la création, la publication et l'exploitation d'applications conteneurisées prêtes à la production sur Amazon ECS à partir d'un environnement de développement local. Le AWS La CLI Copilot s'aligne sur les flux de travail des développeurs qui prennent en charge les meilleures pratiques en matière d'applications modernes : de l'utilisation de l'infrastructure en tant que code à la création d'un CI/CD pipeline provisionné pour le compte d'un utilisateur. Utilisez le AWS Copilot CLI dans le cadre de votre cycle quotidien de développement et de test en tant qu'alternative à Console de gestion AWS.

AWS Copilot est actuellement compatible avec les systèmes Linux, macOS et Windows. Pour plus d'informations sur la dernière version du AWS Copilot CLI, voir Communiqués.

Note

Le code source du AWS L'interface de ligne de commande Copilot est disponible sur GitHub. Nous vous conseillons de soumettre les problèmes et d'extraire les requêtes pour les modifications que vous souhaitez inclure. Toutefois, Amazon Web Services ne prend actuellement pas en charge l'exécution de copies modifiées du code d' AWS Copilot. Signaler des problèmes avec AWS Copilotez en vous connectant avec nous sur Glitter ou GitHub où vous pouvez ouvrir des problèmes, fournir des commentaires et signaler des bogues.

Pour plus d'informations sur le développement de Copilot, consultez la section Quel est le futur de Copilot ?.

Rubriques

- Installation de la AWS Interface de ligne de commande Copilot
- Déploiement d'un exemple d'application Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot

Documentation supplémentaire pour AWS L'interface de ligne de commande Copilot est disponible sur le AWS Site web de Copilot

Installation de la AWS Interface de ligne de commande Copilot

⚠ AWS Interface de ligne de commande Copilot End-of-Support Remarque

AWS La CLI Copilot atteindra la fin du support le 12 juin 2026. Après cette date, l'outil ne bénéficiera plus de mises à jour, de correctifs de sécurité ou de support technique.

- Aucune nouvelle fonctionnalité ou amélioration après le 12 juin 2026
- Aucune mise à jour de sécurité ni correction de bogues
- Pas de support technique
- Les déploiements existants continueront de fonctionner, mais sans maintenance continue

Vous avez besoin d'aide ? Contacter AWS Soutenez ou visitez notre billet de blog pour obtenir des conseils de migration détaillés.

Vous pouvez installer le AWS Copilez la CLI à l'aide de Homebrew ou en téléchargeant manuellement le fichier binaire en suivant les étapes suivantes.

Utilisation de Homebrew

La commande suivante est utilisée pour installer le AWS Copilez la CLI sur votre système macOS ou Linux à l'aide de Homebrew. Avant l'installation, Homebrew doit être installé. Pour plus d'informations, consultez Homebrew.

```
brew install aws/tap/copilot-cli
```

Téléchargement du fichier binaire

Comme alternative à Homebrew, vous pouvez installer manuellement le AWS Copilot CLI sur votre système macOS, Windows ou Linux. Utilisez la commande suivante pour que votre système d'exploitation télécharge le fichier binaire. Les exemples macOS et Linux incluent également des commandes qui appliquent des permissions d'exécution au fichier binaire et affichent le menu d'aide afin de vérifier que l'installation fonctionne.

macOS

Pour Mac OS :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Pour les systèmes macOS ARM :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-darwin-arm64 \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Linux

Pour les systèmes Linux x86 (64 bits) :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux \
   && sudo chmod +x /usr/local/bin/copilot \
   && copilot --help
```

Pour les systèmes ARM Linux :

```
sudo curl -Lo /usr/local/bin/copilot https://github.com/aws/copilot-cli/releases/
latest/download/copilot-linux-arm64 \
```

```
    && sudo chmod +x /usr/local/bin/copilot \
    && copilot --help
```

Windows

Exécutez la commande suivante à l'aide de PowerShell :

```
New-Item -Path 'C:\copilot' -ItemType directory; `
  Invoke-WebRequest -OutFile 'C:\copilot\copilot.exe' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe
```

(Facultatif) Vérifiez que l'installation est manuelle AWS CLI Copilot à l'aide de signatures PGP

Le AWS Les exécutables Copilot CLI sont signés cryptographiquement à l'aide de signatures PGP. Les signatures PGP peuvent être utilisées pour vérifier la validité du AWS Exécutable Copilot CLI. Utilisez les étapes suivantes pour vérifier les signatures à l'aide de l'outil GnuPG.

1. Téléchargez et installez GnuPG. Pour plus d'informations, consultez le site web GnuPG.

   macOS

   Nous vous recommandons d'utiliser Homebrew. Installez Homebrew en suivant les instructions fournies sur leur site web. Pour plus d'informations, consultez Homebrew. Lorsqu'Homebrew est installé, utilisez la commande suivante à partir de votre terminal macOS.

   ```
   brew install gnupg
   ```

   Linux

   Installez gpg à l'aide du gestionnaire de packages sur votre version de Linux.

   Windows

   Téléchargez le programme d'installation Windows simple à partir du site web GnuPG et installez-le en tant qu'administrateur. Après avoir installé GnuPG, fermez et rouvrez l'administrateur PowerShell.

   Pour plus d'informations, consultez Téléchargement de GnuPG.

2. Vérifiez que le chemin GnuPG est ajouté à votre chemin d'environnement.

macOS

```
echo $PATH
```

Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

```
PATH=$PATH:<path to GnuPG executable files>
```

Linux

```
echo $PATH
```

Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

```
export PATH=$PATH:<path to GnuPG executable files>
```

Windows

```
Write-Output $Env:PATH
```

Si vous ne voyez pas le chemin GnuPG dans la sortie, exécutez la commande suivante pour l'ajouter au chemin.

```
$Env:PATH += ";<path to GnuPG executable files>"
```

3. Créez un fichier texte brut local.

macOS

Sur le terminal, saisissez :

```
touch <public_key_filename.txt>
```

Ouvrez le fichier avec TextEdit.

Linux

Créez un fichier texte dans un éditeur de texte, comme gedit. Enregistrer sous `public_key_filename.txt`

Windows

Créez un fichier texte dans un éditeur de texte, comme Notepad. Enregistrer sous `public_key_filename.txt`

4. Ajoutez les éléments de clé publique PGP Amazon ECS suivants et enregistrez le fichier.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----
Version: GnuPG v2

mQINBFq1SasBEADliGcT1NVJ1ydfN8DqebYYe9ne3dt6jqKFmKowLmm6LLGJe7HU
jGtqhCWRDkN+qPpHqdArRgDZAtn2pXY5fEipHgar4CP8QgRnRMO2fl74lmavr4Vg
7K/KH8VHlq2uRw32/B94XLEgRbGTMdWFdKuxoPCttBQaMj3LGn6Pe+6xVWRkChQu
BoQAhjBQ+bEm0kNy0LjNgjNlnL3UMAG56t8E3LANIgGgEnpNsB1UwfWluPoGZoTx
N+6pHBJrKIL/1v/ETU4FXpYw2zvhWNahxeNRnoYj3uycHkeliCrw4kj0+skizBgO
2K7oVX8Oc3j5+ZilhL/qDLXmUCb2az5cMM1mOoF8EKX5HaNuq1KfwJxqXE6NNIcO
lFTrT7QwD5fMNld3FanLgv/ZnIrsSaqJOL6zRSq8O4LN1OWBVbndExk2Kr+5kFxn
5lBPgfPgRj5hQ+KTHMa9Y8Z7yUc64BJiN6F9Nl7FJuSsfqbdkvRLsQRbcBG9qxX3
rJAEhieJzVMEUNl+EgeCkxj5xuSkNU7zw2c3hQZqEcrADLV+hvFJktOz9Gm6xzbq
lTnWWCz4xrIWtuEBA2qE+MlDheVd78a3gIsEaSTfQq0osYXaQbvlnSWOoc1y/5Zb
zizHTJIhLtUyls9WisP2s0emeHZicVMfW61EgPrJAiupgc7kyZvFt4YwfwARAQAB
tCRBbWF6b24gRUNTIDxlY3Mtc2VjdXJpdHlAYW1hem9uLmNvbT6JAhwEEAECAAYF
AlrjL0YACgkQHivRXs0TaQrg1g/+JppwPqHnlVPmv7lessB8I5UqZeD6p6uVpHd7
Bs3pcPp8BV7BdRbs3sPLt5bV1+rkqOlw+0gZ4Q/ue/YbWtOAt4qY0OcEo0HgcnaX
lsB827QIfZIVtGWMhuh94xzm/SJkvngml6KB3YJNnWP61A9qJ37/VbVVLzvcmazA
McWB4HUMNrhd0JgBCo0gIpqCbpJEvUc02Bjn23eEJsS9kC7OUAHyQkVnx4d9UzXF
4OoISF6hmQKIBoLnRrAlj5Qvs3GhvHQ0ThYq0Grk/KMJJX2CSqt7tWJ8gk1n3H3Y
SReRXJRnv7DsDDBwFgT6r5Q2HW1TBUvaoZy5hF6maD09nHcNnvBjqADzeT8Tr/Qu
bBCLzkNSYqqkpgtwv7seoD2P4n1giRvDAOEfMZpVkUr+C252IaH1HZFEz+TvBVQM
Y8OWWxmIJW+J6evjo3N1eO19UHv71jvoF8zljbI4bsL2c+QTJmOv7nRqzDQgCWyp
Id/v2dUVVTk1j9omuLBBwNJzQCB+72LcIzJhYmaP1HC4LcKQG+/f41exuItenatK
lEJQhYtyVXcBlh6Yn/wzNg2NWOwb3vqY/F7m6u9ixAwgtIMgPCDE4aJ86zrrXYFz
N2HqkTSQh77Z8KPKmyGopsmN/reMuilPdINb249nA0dzoN+nj+tTFOYCIaLaFyjs
Z0r1QAOJAjkEEwECACMFAlq1SasCGwMHCwkIBwMCAQYVCAIJCgsEFgIDAQIeAQIX
gAAKCRC86dmkLVF4T9iFEACEnkm1dNXsWUx34R3c0vamHrPxvfkyI1FlEUen8D1h
uX9xy6jCEROHWEp0rjGK4QDPgM93sWJ+s1UAKg214QRVzft0y9/DdR+twApA0fzy
uavIthGd6+03jAAo6udYDE+cZC3P7XBbDiYEWk4XAF9I1JjB8hTZUgvXBL046JhG
eM17+crgUyQeetkiOQemLbsbXQ40Bd9V7zf7XJraFd8VrwNUwNb+9KFtgAsc9rk+
YIT/PEf+YOPysgcxI4sTWghtyCulVnuGoskgDv4v73PALU0ieUrvvQVqWMRvhVx1
```

```
0X90J7cC1KOyhlEQQ1aFTgmQjmXexVTwIBm8LvysFK6YXM41KjOrlz3+6xBIm/qe
bFyLUnf4WoiuOplAaJhK9pRY+XEnGNxdtN4D26Kd0F+PLkm3Tr3Hy3b1Ok34FlGr
KVHUq1TZD7cvMnnNKEELTUcKX+1mV3an16nmAg/my1JSUt6BNK2rJpY1s/kkSGSE
XQ4zuF2IGCpvBFhYAlt5Un5zwqkwwQR3/n2kwAoDzonJcehDw/C/cGos5D0aIU7I
K2X2aTD3+pA7Mx3IMe2hqmYqRt9X42yF1PIEVRneBRJ3HDezAgJrNh0GQWRQkhIx
gz6/cTR+ekr5TptVszS9few2GpI5bCgBKBisZIssT89aw7mAKWut0Gcm4qM9/yK6
1bkCDQRatUmrARAAxNPvVwreJ2yAiFcUpdRlVhsuOgnxvs1QgsIw3H7+Pacr9Hpe
8uftYZqdC82KeSKhpHq7c8gMTMucIINtH25x9BCc73E33EjCL9Lqov1TL7+QkgHe
T+JIhZwdD8Mx2K+LVVVu/aWkNrfMuNwyDUciSI4D5QHa8T+F8fgN4OTpwYjirzel
5yoICMr9hVcbzDNv/ozKCxjx+XKgnFc3wrnDfJfntfDAT7ecwbUTL+viQKJ646s+
psiqXRYtVvYInEhLVrJ0aV6zHFoigE/Bils6/g7ru1Q6CEHqEw++APs5CcE8VzJu
WAGSVHZgun5Y9N4quR/M9Vm+IPMhTxrAg7rOvyRN9cAXfeSMf77I+XTifigNna8x
t/MOdjXr1fjF4pThEi5u6WsuRdFwjY2azEv3vevodTi4HoJReH6dFRa6y8c+UDgl
2iHiOKIpQqLbHEfQmHcDd2fix+AaJKMnPGNku9qCFEMbgSRJpXz6BfwnY1QuKE+I
R6jA0frUNt2jhiGG/F8RceXzohaaC/Cx7LUCUFWc0n7z32C9/Dtj7I1PMOacdZzz
bjJzRKO/ZDv+UN/c9dwAkllzAyPMwGBkUaY68EBstnIliW34aWm6IiHhxioVPKSp
VJfyiXPO0EXqujtHLAeChfjcns3I12YshT1dv2PafG53fp33ZdzeUgsBo+EAEQEA
AYkCHwQYAQIACQUCWrVJqwIbDAAKCRC86dmkLVF4T+ZdD/9x/8APzgNJF3o3STrF
jvnV1ycyhWYGAeBJiu7wjsNWwzMFOv15tLjB7AqeVxZn+WKDD/mIOQ45OZvnYZuy
X7DR0JszaH9wrYTxZLVruAu+t6UL0y/XQ4L1GZ9QR6+r+7t1Mvbfy7BlHbvX/gYt
Rwe/uwdibI0CagEzyX+2D3kTOlHO5XThbXaNf8AN8zha91Jt2Q2UR2X5T6JcwtMz
FBvZnl3LSmZyE0EQehS2iUurU4uWOpGppuqVnbi0jbCvCHKgDGrqZ0smKNAQng54
F365W3g8AfY48s8XQwzmcliowYX9bT8PZiEi0J4QmQh0aXkpqZyFefuWeOL2R94S
XKzr+gRh3BAULoqF+qK+IUMxTip9KTPNvYDpiC66yBiT6gFDji5Ca9pGpJXrC3xe
TXiKQ8DBWDhBPVPrruLIaenTtZEOsPc4I85yt5U9RoPTStcOr34s3w5yEaJagt6S
Gc5r9ysjkfH6+6rbi1ujxMgROSqtqr+RyB+V9A5/OgtNZc8llK6u4UoOCde8jUUW
vqWKvjJB/Kz3u4zaeNu2ZyyHaOqOuH+TETcW+jsY9IhbEzqN5yQYGi4pVmDkY5vu
lXbJnbqPKpRXgM9BecV9AMbPgbDq/5LnHJJXg+G8YQOgp4lR/hC1TEFdIp5wM8AK
CWsENyt2o1rjgMXiZOMF8A5oBLkCDQRatUuSARAAr77kj7j2QR2SZeOSlFBvV7oS
mFeSNnz9xZssqrsm6bTwSHM6YLDwc7Sdf2esDdyzONETwqrVCg+FxgL8hmo9hS4c
rR6tmrP0mOmptr+xLLsKcaP7ogIXsyZnrEAEsvW8PnfayoiPCdc3cMCR/lTnHFGA
7EuR/XLBmi7Qg9tByVYQ5Yj5wB9V4B2yeCt3XtzPqeLKvaxl7PNelaHGJQY/xo+m
V0bndxf9IY+4oFJ4blD32WqvyxESo7vW6WBh7oqv3Zbm0yQrr8a6mDBpqLkvWwNI
3kpJR974tg5o5LfDu1BeeyHWPSGm4U/G4JB+JIG1ADy+RmoWEt4BqTCZ/knnoGvw
D5sTCxbKdmuOmhGyTssoG+3OOcGYHV7pWYPhazKHMPm201xKCjH1RfzRULzGKjD+
yMLT1I3AXFmLmZJXikAOlvE3/wgMqCXscbycbLjLD/bXIuFWo3rzoezeXjgi/DJx
jKBAyBTYO5nMcth1O9oaFd9d0HbsOUDkIMnsgGBE766Piro6MHo0T0rXl07Tp4pI
rwuSOsc6XzCzdImj0Wc6axS/HeUKRXWdXJwno5awTwXKRJMXGfhCvSvbcbc2Wx+L
IKvmB7EB4K3fmjFFE67yolmiw2qRcUBfygtH3eL5XZU28MiCpue8Y8GKJoBAUyvf
KeM1rO8Jm3iRAc5a/D0AEQEAAYkEPgQYAQIACQUCWrVLkgIbAgIpCRC86dmkLVF4
T8FdIAQZAQIABgUCWrVLkgAKCRDePL1hra+LjtHYD/9MucxdFe6bXO1dQR4tKhhQ
P0LRqy6zlBY9ILCLowNdGZdqorogUiUymgn3VhEhVtxTOoHcN7qOuM01PNsRnOeS
EYjf8Xrb1clzkD6xULwmOclTb9bBxnBc/4PFvHAbZW3QzusaZniNgkuxt6BTfloS
Of4inq71kjmGK+TlzQ6mUMQUg228NUQC+a84EPqYyAeY1sgvgB7hJBhYL0QAxhcW
```

```
6m20Rd8iEc6HyzJ3yCOCsKip/nRWAbf0OvfHfRBp0+m0ZwnJM8cPRFjOqqzFpKH9
HpDmTrC4wKP1+TL52LyEqNh4yZitXmZNV7giSRIkk0eDSko+bFy6VbMzKUMkUJK3
D3eHFAMkujmbfJmSMTJOPGn5SB1HyjCZNx6bhIIbQyEUB9gKCmUFaqXKwKpF6rj0
iQXAJxLR/shZ5Rk96VxzOphUl7T90m/PnUEEPwq8KsBhnMRgxa0RFidDP+n9fgtv
HLmrOqX9zBCVXh0mdWYLrWvmzQFWzG7AoE55fkf8nAEPsalrCdtaNUBHRXA0OQxG
AHMOdJQQvBsmqMvuAdjkDWpFu5y0My5ddU+hiUzUyQLjL5Hhd5LOUDdewlZgIw1j
xrEAUzDKetnemM8GkHxDgg8koev5frmShJuce7vSjKpCNg3EIJSgqMOPFjJuLWtZ
vjHeDNbJy6uNL65ckJy6WhGjEADS2WAW1D6Tfekkc21SsIXk/LqEpLMR/0g5OUif
wcEN1rS9IJXBwIy8MelN9qr5KcKQLmfdfBNEyyceBhyVl0MDyHOKC+7PofMtkGBq
13QieRHv5GJ8LB3fclqHV8pwTTo3Bc8z2g0TjmUYAN/ixETdReDoKavWJYSE9yoM
aaJu279ioVTrwpECse0XkiRyKToTjwOb73CGkBZZpJyqux/rmCV/fp4ALdSW8zbz
FJVORaivhoWwzjpfQKhwcU9lABXi2UvVm14v0AfeI7oiJPSU1zM4fEny4oiIBXlR
zhFNih1UjIu82X16mTm3BwbIga/s1fnQRGzyhqUIMii+mWra23EwjChaxpvjjcUH
5ilLc5Zq781aCYRygYQw+hu5nFkOH1R+Z50Ubxjd/aqUfnGIAX7kPMD3Lof4KldD
Q8ppQriUvxVo+4nPV6rpTy/PyqCLWDjkguHpJsEFsMkwajrAz0QNSAU5CJ0G2Zu4
yxvYlumHCEl7nbFrm0vIiA75Sa8KnywTDsyZsu3XcOcf3g+g1xWTpjJqy2bYXlqz
9uDOWtArWHOis6bq8l9RE6xr1RBVXS6uqgQIZFBGyq66b0dIq4D2JdsUvgEMaHbc
e7tBfeB1CMBdA64e9Rq7bFR7Tvt8gasCZYlNr3lydh+dFHIEkH53HzQe6l88HEic
+0jVnLkCDQRa55wJARAAyLya2Lx6gyoWoJN1a6740q3o8e9d4KggQOfGMTCflmeq
ivuzgN+3DZHN+9ty2KxXMtn0mhHBerZdbNJyjMNT1gAgrhPNB4HtXBXum2wS57WK
DNmade914L7FWTPAWBG2Wn448OEHTqsClICXXWy9IICgclAEyIq0Yq5mAdTEgRJS
Z8t4GpwtDL9gNQyFXaWQmDmkAsCygQMvhAlmu9xOIzQG5CxSnZFk7zcuL60k14Z3
Cmt49k4T/7ZU8goWi8tt+rU78/IL3J/fF9+1civ1OwuUidgfPCSvOUW1JojsdCQA
L+RZJcoXq7lfOFj/eNjeOSstCTDPfTCL+kThE6E5neDtbQHBYkEX1BRiTedsV4+M
ucgiTrdQFWKf89G72xdv8ut9AYYQ2BbEYU+JAYhUH8rYYui2dHKJIgjNvJscuUWb
+QEqJIRleJRhrO+/CHgMs4fZAkWF1VFhKBkcKmEjLn1f7EJJUUW84ZhKXjO/AUPX
1CHsNjziRceuJCJYox1cwsoq6jTE50GiNzcIxTn9xUc0UMKFeggNAFys1K+TDTm3
Bzo8H5ucjCUEmUm9lhkGwqTZgOlRX5eqPX+JBoSaObqhgqCa5IPinKRa6MgoFPHK
6sYKqroYwBGgZm6Js5chpNchvJMs/3WXNOEVg0J3z3vP0DMhxqWm+r+n9zlW8qsA
EQEAAYkEPgQYAQgACQUCWuecCQIbAgIpCRC86dmkLVF4T8FdIAQZAQgABgUCWuec
CQAKCRBQ3szEcQ5hr+ykD/4tOLRHFHXuKUcxgGaubUcVtsFrwBKma1cYjqaPms8u
6Sk0wfGRI32G/GhOrp0Ts/MOkbObq6VLTh8N5Yc/53MEl8zQFw9Y5AmRoW4PZXER
ujs5s7p4oR7xHMihMjCCBn1bvrR+34YPfgzTcgLiOEFHYT8UTxwnGmXOvNkMM7md
xD3CV5q6VAte8WKBo/220II3fcQlc9r/oWX4kXXkb0v9hoGwKbDJ1tzqTPrp/xFt
yohqnvImpnlz+Q9zXmbrWYL9/g8VCmW/NN2gju2G3Lu/TlFUWIT4v/5OPK6TdeNb
VKJO4+S8bTayqSG9CML1S57KSgCo5HUhQWeSNHI+fpe5oX6FALPT9JLDce8OZz1i
cZZ0MELP37mOOQun0AlmHm/hVzf0f311PtbzcqWaE51tJvgUR/nZFo6Ta3O5Ezhs
3VlEJNQ1Ijf/6DH87SxvAoRIARCuZd0qxBcDK0avpFzUtbJd24lRA3WJpkEiMqKv
RDVZkE4b6TW61f0o+LaVfK6E8oLpixegS4fiqC16mFrOdyRk+RJJfIUyz0WTDVmt
g0U1CO1ezokMSqkJ7724pyjr2xf/r9/sC6aOJwB/lKgZkJfC6NqL7TlxVA31dUga
LEOvEJTTE4gl+tYtfsCDvALCtqL0jduSkUo+RXcBItmXhA+tShW0pbS2Rtx/ixua
KohVD/0R4QxiSwQmICNtm9mw9ydIl1yjYXX5a9x4wMJracNY/LBybJPFnZnT4dYR
z4XjqysDwvvYZByaWoIe3QxjX84V6MlI2IdAT/xImu8gbaCI8tmyfpIrLnPKiR9D
VFYfGBXuAX7+HgPPSFtrHQONCALxxzlbNpS+zxt9r0MiLgcLyspWxSdmoYGZ6nQP
```

```
RO5Nm/ZVS+u2imPCRzNUZEMa+dlE6kHx0rS0dPiuJ4O7NtPeYDKkoQtNagspsDvh
cK7CSqAiKMq06UBTxqlTSRkm62eOCtcs3p3OeHu5GRZF1uzTET0ZxYkaPgdrQknx
ozjP5mC7X+45lcCfmcVt94TFNL5HwEUVJpmOgmzILCI8yoDTWzloo+i+fPFsXX4f
kynhE83mSEcr5VHFYrTY3mQXGmNJ3bCLuc/jq7ysGq69xiKmTlUeXFm+aojcRO5i
zyShIRJZ0GZfuzDYFDbMV9amA/YQGygLw//zP5ju5SW26dNxlf3MdFQE5JJ86rn9
MgZ4gcpazHEVUsbZsgkLizRp9imUiH8ymLqAXnfRGlU/LpNSefnvDFTtEIRcpOHc
bhayG0bk51Bd4mioOXnIsKy4j63nJXA27x5EVVHQ1sYRN8Ny4Fdr2tMAmj2O+X+J
qX2yy/UX5nSPU492e2CdZ1UhoU0SRFY3bxKHKB7SDbVeav+K5g==
=Gi5D
-----END PGP PUBLIC KEY BLOCK-----
```

Détails de la clé publique PGP Amazon ECS pour référence :

```
Key ID: BCE9D9A42D51784F
Type: RSA
Size: 4096/4096
Expires: Never
User ID: Amazon ECS
Key fingerprint: F34C 3DDA E729 26B0 79BE AEC6 BCE9 D9A4 2D51 784F
```

5. Importez le fichier avec la clé publique PGP Amazon ECS avec la commande suivante dans le terminal.

```
gpg --import <public_key_filename.txt>
```

6. Téléchargez le AWS Signatures CLI du copilote. Les signatures sont des signatures PGP détachées ASCII, stockées dans des fichiers portant l'extension `.asc`. Le fichier de signatures a le même nom que son ficher exécutable correspondant, avec l'extension `.asc`.

   macOS

   Pour les systèmes macOS, exécutez la commande suivante :

   ```
   sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/download/copilot-darwin.asc
   ```

   Linux

   Pour les systèmes Linux x86 (64 bits), exécutez la commande suivante :

```
sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
download/copilot-linux.asc
```

Pour les systèmes ARM Linux, exécutez la commande suivante :

```
sudo curl -Lo copilot.asc https://github.com/aws/copilot-cli/releases/latest/
download/copilot-linux-arm64.asc
```

Windows

Exécutez la commande suivante à l'aide de PowerShell.

```
Invoke-WebRequest -OutFile 'C:\copilot\copilot.asc' https://github.com/aws/
copilot-cli/releases/latest/download/copilot-windows.exe.asc
```

7. Vérifiez la signature avec la commande suivante :

   - Pour les systèmes Mac OS et Linux :

     ```
     gpg --verify copilot.asc /usr/local/bin/copilot
     ```

   - Pour les systèmes Windows :

     ```
     gpg --verify 'C:\copilot\copilot.asc' 'C:\copilot\copilot.exe'
     ```

   Sortie attendue :

   ```
   gpg: Signature made Tue Apr  3 13:29:30 2018 PDT
   gpg:                using RSA key DE3CBD61ADAF8B8E
   gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
   gpg: WARNING: This key is not certified with a trusted signature!
   gpg:          There is no indication that the signature belongs to the owner.
   Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
        Subkey fingerprint: EB3D F841 E2C9 212A 2BD4  2232 DE3C BD61 ADAF 8B8E
   ```

> **Important**
>
> L'avertissement présent dans la sortie est prévu et ne constitue pas un problème. Cet avertissement est déclenché, car il n'y a pas de chaîne de confiance entre votre clé PGP personnelle (si vous en avez une) et la clé PGP Amazon ECS. Pour de plus amples informations, consultez Web of trust.

8. Pour les installations Windows, exécutez la commande suivante sur Powershell pour ajouter AWS Copilotez le répertoire vers le chemin.

```
$Env:PATH += ";<path to Copilot executable files>"
```

Déploiement d'un exemple d'application Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot

> **AWS Interface de ligne de commande Copilot End-of-Support Remarque**
>
> AWS La CLI Copilot atteindra la fin du support le 12 juin 2026. Après cette date, l'outil ne bénéficiera plus de mises à jour, de correctifs de sécurité ou de support technique.
>
> - Aucune nouvelle fonctionnalité ou amélioration après le 12 juin 2026
> - Aucune mise à jour de sécurité ni correction de bogues
> - Pas de support technique
> - Les déploiements existants continueront de fonctionner, mais sans maintenance continue
>
> Vous avez besoin d'aide ? Contacter AWS Soutenez ou visitez notre billet de blog pour obtenir des conseils de migration détaillés.

Après avoir installé le AWS Copilot CLI, vous pouvez suivre ces étapes pour déployer un exemple d'application, vérifier le déploiement et nettoyer les ressources.

Conditions préalables

Avant de commencer, assurez-vous de remplir les prérequis suivants :

- Installez et configurez l' AWS CLI. Pour plus d'informations, consultez Interface de ligne de commande AWS.
- Courez `aws configure` pour configurer un profil par défaut que le AWS La CLI Copilot sera utilisée pour gérer votre application et vos services.
- Installez et exécutez Docker. Pour plus d'informations, veuillez consulter Get started with Docker.

Déploiement d'un exemple d'application Amazon ECS en une seule commande

1. Déployez un exemple d'application Web cloné à partir d'un GitHub référentiel à l'aide de la commande suivante. Pour plus d'informations sur AWS Copilote `init` et ses drapeaux, voir le AWS Documentation du copilote.

```
git clone https://github.com/aws-samples/aws-copilot-sample-service.git demo-app &&
 \
cd demo-app &&                                \
copilot init --app demo                       \
  --name api                                  \
  --type 'Load Balanced Web Service'          \
  --dockerfile './Dockerfile'                 \
  --port 80                                   \
  --tag  latest                               \
  --deploy
```

2. Une fois le déploiement terminé, le AWS L'interface de ligne de commande Copilot renverra une URL que vous pourrez utiliser pour vérifier le déploiement. Vous pouvez également utiliser les commandes suivantes pour vérifier l'état de l'application.

   - Dressez la liste de tous vos AWS Applications de copilote.

   ```
   copilot app ls
   ```

   - Affichez des informations sur les environnements et les services dans votre application.

   ```
   copilot app show
   ```

   - Affichez des informations sur vos environnements.

   ```
   copilot env ls
   ```

- Affiche des informations sur le service, y compris les points de terminaison, la capacité et les ressources connexes.

```
copilot svc show
```

- Liste de tous les services d'une application.

```
copilot svc ls
```

- Affichez les journaux d'un service déployé.

```
copilot svc logs
```

- Affichez l'état du service.

```
copilot svc status
```

3. Une fois cette démonstration terminée, exécutez la commande suivante pour nettoyer les ressources associées et éviter les frais liés aux ressources inutilisées.

```
copilot app delete
```

Utilisation d'Amazon ECS avec AWS CloudFormation

Amazon ECS est intégré à AWS CloudFormation un service que vous pouvez utiliser pour modéliser et configurer des AWS ressources à l'aide de modèles que vous définissez. CloudFormation utilise des modèles sous forme de fichier texte `YAML` ou de fichier texte `JSON` formaté. Les modèles sont comme des plans pour la AWS ressource que vous souhaitez créer. Lorsque vous créez et soumettez un modèle, CloudFormation crée une pile. La pile vous permet de gérer les ressources que vous avez définies dans votre modèle. Lorsque vous voulez créer, mettre à jour ou supprimer une ressource, vous devez créer, mettre à jour ou supprimer la pile qui a été créée à partir de cette ressource. Lorsqu'il s'agit de mettre à jour vos piles, vous devez d'abord créer un jeu de modifications. Les jeux de modifications vous indiquent ce qui est impacté par la modification avant que vous ne l'apportiez. Cela vous empêche de supprimer des bases de données accidentellement en modifiant le nom de votre base de données, par exemple. Pour plus d'informations sur les modèles, les piles et les ensembles de modifications, consultez la section CloudFormation Fonctionnement du guide de l'AWS CloudFormation utilisateur.

En l'utilisant CloudFormation, vous pouvez passer moins de temps à créer et à gérer vos ressources et votre infrastructure. Vous pouvez créer un modèle qui décrit toutes les AWS ressources souhaitées, telles que les clusters Amazon ECS, les définitions de tâches et les services. CloudFormation Il s'occupe ensuite du provisionnement et de la configuration de ces ressources pour vous.

CloudFormation vous permet également de réutiliser votre modèle pour configurer vos ressources Amazon ECS de manière cohérente et reproductible. Vous décrivez vos ressources une seule fois, puis vous réapprovisionnez les mêmes ressources sur plusieurs Comptes AWS et Régions AWS.

CloudFormation les modèles peuvent être utilisés à la fois avec Console de gestion AWS ou AWS Command Line Interface pour créer des ressources.

Pour en savoir plus CloudFormation, consultez les ressources suivantes :

- AWS CloudFormation
- AWS CloudFormation Guide de l'utilisateur
- AWS CloudFormation Guide de l'utilisateur de l'interface de ligne de commande

Rubriques

- Création de ressources Amazon ECS à l'aide de la CloudFormation console

- Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour CloudFormation
- CloudFormation exemples de modèles pour Amazon ECS

Création de ressources Amazon ECS à l'aide de la CloudFormation console

L'une des manières d'utiliser Amazon ECS CloudFormation est d'utiliser le Console de gestion AWS. Ici, vous pouvez créer vos piles CloudFormation pour les composants Amazon ECS tels que les définitions de tâches, les clusters et les services et les déployer directement depuis la console. Le didacticiel suivant montre comment utiliser la console CloudFormation pour créer des ressources Amazon ECS à l'aide d'un modèle.

Conditions préalables

Ce didacticiel suppose que les conditions préalables suivantes ont été remplies.

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.

Étape 1 : créer un modèle

Suivez les étapes ci-dessous pour créer un modèle de CloudFormation pile pour un service Amazon ECS et d'autres ressources associées.

1. À l'aide d'un éditeur de texte de votre choix, créez un fichier appelé `ecs-tutorial-template.yaml`.
2. Dans le fichier `ecs-tutorial-template.yaml`, collez le modèle suivant et enregistrez les modifications.

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] ECS: load-balanced-web-application'

Parameters:
  VpcCidr:
    Type: String
    Default: '10.0.0.0/16'
```

```
    Description: CIDR block for the VPC
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest'
    Description: Container image to use in task definition

  PublicSubnet1Cidr:
    Type: String
    Default: '10.0.1.0/24'
    Description: CIDR block for public subnet 1

  PublicSubnet2Cidr:
    Type: String
    Default: '10.0.2.0/24'
    Description: CIDR block for public subnet 2

  PrivateSubnet1Cidr:
    Type: String
    Default: '10.0.3.0/24'
    Description: CIDR block for private subnet 1

  PrivateSubnet2Cidr:
    Type: String
    Default: '10.0.4.0/24'
    Description: CIDR block for private subnet 2

  ServiceName:
    Type: String
    Default: 'tutorial-app'
    Description: Name of the ECS service

  ContainerPort:
    Type: Number
    Default: 80
    Description: Port on which the container listens

  DesiredCount:
    Type: Number
    Default: 2
    Description: Desired number of tasks

  MinCapacity:
    Type: Number
    Default: 1
```

```
    Description: Minimum number of tasks for auto scaling

  MaxCapacity:
    Type: Number
    Default: 10
    Description: Maximum number of tasks for auto scaling

Resources:
  # VPC and Networking
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsHostnames: true
      EnableDnsSupport: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-vpc'

  # Internet Gateway
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-igw'

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  # Public Subnets for ALB
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-1'
```

```
  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-2'

  # Private Subnets for ECS Tasks
  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet1Cidr
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-subnet-1'

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet2Cidr
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-subnet-2'

  # NAT Gateways for private subnet internet access
  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-1'

  NatGateway2EIP:
```

```
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-2'

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-1'

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-2'

  # Route Tables
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-routes'

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
```

```
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2

  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-1'

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-2'

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
```

```
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  # Security Groups
  ALBSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${AWS::StackName}-alb-sg'
      GroupDescription: Security group for Application Load Balancer
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
          Description: Allow HTTP traffic from internet
      SecurityGroupEgress:
        - IpProtocol: -1
          CidrIp: 0.0.0.0/0
          Description: Allow all outbound traffic
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-alb-sg'

  ECSSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${AWS::StackName}-ecs-sg'
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: !Ref ContainerPort
          ToPort: !Ref ContainerPort
          SourceSecurityGroupId: !Ref ALBSecurityGroup
          Description: Allow traffic from ALB
      SecurityGroupEgress:
        - IpProtocol: -1
```

```
          CidrIp: 0.0.0.0/0
          Description: Allow all outbound traffic
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-ecs-sg'

  # Application Load Balancer
  ApplicationLoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Sub '${AWS::StackName}-alb'
      Scheme: internet-facing
      Type: application
      Subnets:
        - !Ref PublicSubnet1
        - !Ref PublicSubnet2
      SecurityGroups:
        - !Ref ALBSecurityGroup
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-alb'

  ALBTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      Name: !Sub '${AWS::StackName}-tg'
      Port: !Ref ContainerPort
      Protocol: HTTP
      VpcId: !Ref VPC
      TargetType: ip
      HealthCheckIntervalSeconds: 30
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 5
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-tg'

  ALBListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
```

```
      - Type: forward
        TargetGroupArn: !Ref ALBTargetGroup
    LoadBalancerArn: !Ref ApplicationLoadBalancer
    Port: 80
    Protocol: HTTP

# ECS Cluster
ECSCluster:
  Type: AWS::ECS::Cluster
  Properties:
    ClusterName: !Sub '${AWS::StackName}-cluster'
    CapacityProviders:
      - FARGATE
      - FARGATE_SPOT
    DefaultCapacityProviderStrategy:
      - CapacityProvider: FARGATE
        Weight: 1
      - CapacityProvider: FARGATE_SPOT
        Weight: 4
    ClusterSettings:
      - Name: containerInsights
        Value: enabled
    Tags:
      - Key: Name
        Value: !Sub '${AWS::StackName}-cluster'

# IAM Roles
ECSTaskExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    RoleName: !Sub '${AWS::StackName}-task-execution-role'
    AssumeRolePolicyDocument:
      Version: '2012-10-17       '
      Statement:
        - Effect: Allow
          Principal:
            Service: ecs-tasks.amazonaws.com
          Action: sts:AssumeRole
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
    Tags:
      - Key: Name
        Value: !Sub '${AWS::StackName}-task-execution-role'
```

```
  ECSTaskRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub '${AWS::StackName}-task-role'
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task-role'

  # CloudWatch Log Group
  LogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: !Sub '/ecs/${AWS::StackName}'
      RetentionInDays: 7

  # ECS Task Definition
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Sub '${AWS::StackName}-task'
      Cpu: '256'
      Memory: '512'
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      TaskRoleArn: !GetAtt ECSTaskRole.Arn
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Image: !Ref ContainerImage
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          Essential: true
          LogConfiguration:
            LogDriver: awslogs
            Options:
```

```
            awslogs-group: !Ref LogGroup
            awslogs-region: !Ref AWS::Region
            awslogs-stream-prefix: ecs
        HealthCheck:
          Command:
            - CMD-SHELL
            - curl -f http://localhost/ || exit 1
          Interval: 30
          Timeout: 5
          Retries: 3
          StartPeriod: 60
    Tags:
      - Key: Name
        Value: !Sub '${AWS::StackName}-task'

# ECS Service
ECSService:
  Type: AWS::ECS::Service
  DependsOn: ALBListener
  Properties:
    ServiceName: !Sub '${AWS::StackName}-service'
    Cluster: !Ref ECSCluster
    TaskDefinition: !Ref TaskDefinition
    DesiredCount: !Ref DesiredCount
    LaunchType: FARGATE
    PlatformVersion: LATEST
    NetworkConfiguration:
      AwsvpcConfiguration:
        AssignPublicIp: DISABLED
        SecurityGroups:
          - !Ref ECSSecurityGroup
        Subnets:
          - !Ref PrivateSubnet1
          - !Ref PrivateSubnet2
    LoadBalancers:
      - ContainerName: !Ref ServiceName
        ContainerPort: !Ref ContainerPort
        TargetGroupArn: !Ref ALBTargetGroup
    DeploymentConfiguration:
      MaximumPercent: 200
      MinimumHealthyPercent: 50
      DeploymentCircuitBreaker:
        Enable: true
        Rollback: true
```

```
      EnableExecuteCommand: true  # For debugging
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-service'

  # Auto Scaling Target
  ServiceScalingTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: !Ref MaxCapacity
      MinCapacity: !Ref MinCapacity
      ResourceId: !Sub 'service/${ECSCluster}/${ECSService.Name}'
      RoleARN: !Sub 'arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService'
      ScalableDimension: ecs:service:DesiredCount
      ServiceNamespace: ecs

  # Auto Scaling Policy - CPU Utilization
  ServiceScalingPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub '${AWS::StackName}-cpu-scaling-policy'
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ServiceScalingTarget
      TargetTrackingScalingPolicyConfiguration:
        PredefinedMetricSpecification:
          PredefinedMetricType: ECSServiceAverageCPUUtilization
        TargetValue: 70.0
        ScaleOutCooldown: 300
        ScaleInCooldown: 300

Outputs:
  VPCId:
    Description: VPC ID
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPC-ID'

  LoadBalancerURL:
    Description: URL of the Application Load Balancer
    Value: !Sub 'http://${ApplicationLoadBalancer.DNSName}'
    Export:
      Name: !Sub '${AWS::StackName}-ALB-URL'
```

```
ECSClusterName:
  Description: Name of the ECS Cluster
  Value: !Ref ECSCluster
  Export:
    Name: !Sub '${AWS::StackName}-ECS-Cluster'

ECSServiceName:
  Description: Name of the ECS Service
  Value: !GetAtt ECSService.Name
  Export:
    Name: !Sub '${AWS::StackName}-ECS-Service'

PrivateSubnet1:
  Description: Private Subnet 1 ID
  Value: !Ref PrivateSubnet1
  Export:
    Name: !Sub '${AWS::StackName}-Private-Subnet-1'

PrivateSubnet2:
  Description: Private Subnet 2 ID
  Value: !Ref PrivateSubnet2
  Export:
    Name: !Sub '${AWS::StackName}-Private-Subnet-2'
```

Le modèle utilisé dans ce didacticiel crée un service Amazon ECS avec deux tâches exécutées sur Fargate. Les tâches exécutent chacune un exemple d'application Amazon ECS. Le modèle crée également un Application Load Balancer qui distribue le trafic des applications et une politique Application Auto Scaling qui adapte l'application en fonction de l'utilisation de l'UC. Le modèle crée également les ressources réseau nécessaires au déploiement de l'application, les ressources de journalisation pour les journaux de conteneurs et un rôle IAM d'exécution de tâches Amazon ECS. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS. Pour plus d'informations sur l'autoscaling, consultez la section Mise à l'échelle automatique de votre service Amazon ECS.

Étape 2 : créer une pile pour les ressources Amazon ECS

Après avoir créé un fichier pour le modèle, vous pouvez suivre ces étapes pour créer une pile avec le modèle à l'aide de la CloudFormation console.

Pour plus d'informations sur la création d'une pile à l'aide de la CloudFormation console, consultez la section Création d'une pile sur la CloudFormation console dans le Guide de l'AWS CloudFormation utilisateur et utilisez le tableau suivant pour déterminer les options à spécifier.

Option	Value
Prérequis : préparer le modèle	Choisissez un modèle existant
Spécifier le modèle	Charger un fichier de modèle
Choisissez un fichier	ecs-tutorial-template.yaml
Nom de la pile	ecs-tutorial-stack
Parameters	Laissez toutes les valeurs des paramètres par défaut.
Fonctionnalités	Choisissez Je reconnais que ce modèle peut créer des ressources IAM pour confirmer la CloudFormation création de ressources IAM.

Étape 3 : vérifier

Suivez les étapes suivantes pour vérifier la création de ressources Amazon ECS à l'aide du modèle fourni.

Pour plus d'informations sur la façon d'afficher les informations et les ressources de la pile, consultez la section Affichage des informations de la pile depuis la CloudFormation console dans le Guide de l'AWS CloudFormation utilisateur et utilisez le tableau suivant pour déterminer les éléments à vérifier.

Champ de détails de la pile	Ce qu'il faut rechercher
Stack info (Infos de pile)	Un statut `CREATE_COMPLETE` .
Ressources	Liste des ressources créées avec des liens vers la console

Champ de détails de la pile	Ce qu'il faut rechercher
	de service. Choisissez des liens vers `ECSCluster` , `ECSService` et `TaskDefinition` pour afficher plus de détails sur le service, le cluster et la définition de tâche créés dans la console Amazon ECS.
Sorties	LoadBalancerURL. Collez l'URL dans un navigateur Web pour afficher une page Web qui affiche un exemple d'application Amazon ECS.

Étape 4 : Nettoyer les ressources

Pour nettoyer les ressources et éviter d'encourir des coûts supplémentaires, suivez les étapes décrites dans Supprimer une pile de la CloudFormation console dans le guide de l'CloudFormation utilisateur.

Création de ressources Amazon ECS à l'aide de AWS CLI commandes pour CloudFormation

Une autre façon d'utiliser Amazon ECS CloudFormation consiste à utiliser le AWS CLI. Vous pouvez utiliser des commandes pour créer vos CloudFormation piles pour les composants Amazon ECS tels que les définitions de tâches, les clusters et les services, puis les déployer. Le didacticiel suivant montre comment vous pouvez utiliser le AWS CLI pour créer des ressources Amazon ECS à l'aide d'un CloudFormation modèle.

Conditions préalables

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.

Étape 1 : créer une pile

Pour créer une pile en utilisant le fichier AWS CLI enregistré dans un fichier appeléecs-tutorial-template.yaml, exécutez la commande suivante.

```
cat << 'EOF' > ecs-tutorial-template.yaml
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] ECS: load-balanced-web-application'
Parameters:
  VpcCidr:
    Type: String
    Default: '10.0.0.0/16'
    Description: CIDR block for the VPC
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest'
    Description: Container image to use in task definition

  PublicSubnet1Cidr:
    Type: String
    Default: '10.0.1.0/24'
    Description: CIDR block for public subnet 1

  PublicSubnet2Cidr:
    Type: String
    Default: '10.0.2.0/24'
    Description: CIDR block for public subnet 2

  PrivateSubnet1Cidr:
    Type: String
    Default: '10.0.3.0/24'
    Description: CIDR block for private subnet 1

  PrivateSubnet2Cidr:
    Type: String
    Default: '10.0.4.0/24'
    Description: CIDR block for private subnet 2

  ServiceName:
    Type: String
    Default: 'tutorial-app'
    Description: Name of the ECS service
```

```
  ContainerPort:
    Type: Number
    Default: 80
    Description: Port on which the container listens

  DesiredCount:
    Type: Number
    Default: 2
    Description: Desired number of tasks

  MinCapacity:
    Type: Number
    Default: 1
    Description: Minimum number of tasks for auto scaling

  MaxCapacity:
    Type: Number
    Default: 10
    Description: Maximum number of tasks for auto scaling

Resources:
  # VPC and Networking
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsHostnames: true
      EnableDnsSupport: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-vpc'

  # Internet Gateway
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-igw'

  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
```

```
      VpcId: !Ref VPC

  # Public Subnets for ALB
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-1'

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-2'

  # Private Subnets for ECS Tasks
  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet1Cidr
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-subnet-1'

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet2Cidr
      Tags:
        - Key: Name
```

```
        Value: !Sub '${AWS::StackName}-private-subnet-2'

  # NAT Gateways for private subnet internet access
  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-1'

  NatGateway2EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-2'

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-1'

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-2'

  # Route Tables
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
```

```
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-routes'

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2

  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-1'

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1
```

```
  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-2'

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  # Security Groups
  ALBSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${AWS::StackName}-alb-sg'
      GroupDescription: Security group for Application Load Balancer
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
          Description: Allow HTTP traffic from internet
      SecurityGroupEgress:
        - IpProtocol: -1
          CidrIp: 0.0.0.0/0
          Description: Allow all outbound traffic
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-alb-sg'

  ECSSecurityGroup:
```

```
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${AWS::StackName}-ecs-sg'
      GroupDescription: Security group for ECS tasks
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: !Ref ContainerPort
          ToPort: !Ref ContainerPort
          SourceSecurityGroupId: !Ref ALBSecurityGroup
          Description: Allow traffic from ALB
      SecurityGroupEgress:
        - IpProtocol: -1
          CidrIp: 0.0.0.0/0
          Description: Allow all outbound traffic
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-ecs-sg'

  # Application Load Balancer
  ApplicationLoadBalancer:
    Type: AWS::ElasticLoadBalancingV2::LoadBalancer
    Properties:
      Name: !Sub '${AWS::StackName}-alb'
      Scheme: internet-facing
      Type: application
      Subnets:
        - !Ref PublicSubnet1
        - !Ref PublicSubnet2
      SecurityGroups:
        - !Ref ALBSecurityGroup
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-alb'

  ALBTargetGroup:
    Type: AWS::ElasticLoadBalancingV2::TargetGroup
    Properties:
      Name: !Sub '${AWS::StackName}-tg'
      Port: !Ref ContainerPort
      Protocol: HTTP
      VpcId: !Ref VPC
      TargetType: ip
      HealthCheckIntervalSeconds: 30
```

```
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 5
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-tg'

  ALBListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
        - Type: forward
          TargetGroupArn: !Ref ALBTargetGroup
      LoadBalancerArn: !Ref ApplicationLoadBalancer
      Port: 80
      Protocol: HTTP

  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Sub '${AWS::StackName}-cluster'
      CapacityProviders:
        - FARGATE
        - FARGATE_SPOT
      DefaultCapacityProviderStrategy:
        - CapacityProvider: FARGATE
          Weight: 1
        - CapacityProvider: FARGATE_SPOT
          Weight: 4
      ClusterSettings:
        - Name: containerInsights
          Value: enabled
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-cluster'

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub '${AWS::StackName}-task-execution-role'
```

```
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task-execution-role'

  ECSTaskRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub '${AWS::StackName}-task-role'
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task-role'

  # CloudWatch Log Group
  LogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: !Sub '/ecs/${AWS::StackName}'
      RetentionInDays: 7

  # ECS Task Definition
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Sub '${AWS::StackName}-task'
      Cpu: '256'
      Memory: '512'
      NetworkMode: awsvpc
      RequiresCompatibilities:
```

```
        - FARGATE
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      TaskRoleArn: !GetAtt ECSTaskRole.Arn
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Image: !Ref ContainerImage
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          Essential: true
          LogConfiguration:
            LogDriver: awslogs
            Options:
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          HealthCheck:
            Command:
              - CMD-SHELL
              - curl -f http://localhost/ || exit 1
            Interval: 30
            Timeout: 5
            Retries: 3
            StartPeriod: 60
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task'

  # ECS Service
  ECSService:
    Type: AWS::ECS::Service
    DependsOn: ALBListener
    Properties:
      ServiceName: !Sub '${AWS::StackName}-service'
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref TaskDefinition
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      PlatformVersion: LATEST
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: DISABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
```

```
          Subnets:
            - !Ref PrivateSubnet1
            - !Ref PrivateSubnet2
      LoadBalancers:
        - ContainerName: !Ref ServiceName
          ContainerPort: !Ref ContainerPort
          TargetGroupArn: !Ref ALBTargetGroup
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 50
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
      EnableExecuteCommand: true  # For debugging
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-service'

  # Auto Scaling Target
  ServiceScalingTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: !Ref MaxCapacity
      MinCapacity: !Ref MinCapacity
      ResourceId: !Sub 'service/${ECSCluster}/${ECSService.Name}'
      RoleARN: !Sub 'arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService'
      ScalableDimension: ecs:service:DesiredCount
      ServiceNamespace: ecs

  # Auto Scaling Policy - CPU Utilization
  ServiceScalingPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub '${AWS::StackName}-cpu-scaling-policy'
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ServiceScalingTarget
      TargetTrackingScalingPolicyConfiguration:
        PredefinedMetricSpecification:
          PredefinedMetricType: ECSServiceAverageCPUUtilization
        TargetValue: 70.0
        ScaleOutCooldown: 300
        ScaleInCooldown: 300
```

```
Outputs:
  VPCId:
    Description: VPC ID
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPC-ID'

  LoadBalancerURL:
    Description: URL of the Application Load Balancer
    Value: !Sub 'http://${ApplicationLoadBalancer.DNSName}'
    Export:
      Name: !Sub '${AWS::StackName}-ALB-URL'

  ECSClusterName:
    Description: Name of the ECS Cluster
    Value: !Ref ECSCluster
    Export:
      Name: !Sub '${AWS::StackName}-ECS-Cluster'

  ECSServiceName:
    Description: Name of the ECS Service
    Value: !GetAtt ECSService.Name
    Export:
      Name: !Sub '${AWS::StackName}-ECS-Service'

  PrivateSubnet1:
    Description: Private Subnet 1 ID
    Value: !Ref PrivateSubnet1
    Export:
      Name: !Sub '${AWS::StackName}-Private-Subnet-1'

  PrivateSubnet2:
    Description: Private Subnet 2 ID
    Value: !Ref PrivateSubnet2
    Export:
      Name: !Sub '${AWS::StackName}-Private-Subnet-2'
EOF
```

Le modèle utilisé dans ce didacticiel crée un service Amazon ECS avec deux tâches exécutées sur Fargate. Les tâches exécutent chacune un exemple d'application Amazon ECS. Le modèle crée également un Application Load Balancer qui distribue le trafic des applications et une politique Application Auto Scaling qui adapte l'application en fonction de l'utilisation de l'UC. Le modèle crée

également les ressources réseau nécessaires au déploiement de l'application, les ressources de journalisation pour les journaux de conteneurs et un rôle IAM d'exécution de tâches Amazon ECS. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS. Pour plus d'informations sur l'autoscaling, consultez la section Mise à l'échelle automatique de votre service Amazon ECS.

Après avoir créé un fichier modèle, utilisez la commande suivante pour créer une pile. L'indicateur `--capabilities` est nécessaire pour créer un rôle d'exécution de tâche Amazon ECS tel que spécifié dans le modèle. Vous pouvez également spécifier l'indicateur `--parameters` pour personnaliser les paramètres du modèle.

```
aws cloudformation create-stack \
      --stack-name ecs-tutorial-stack \
      --template-body file://ecs-tutorial-template.yaml \
      --region aws-region \
      --capabilities CAPABILITY_NAMED_IAM
```

Après avoir exécuté la commande `create-stack`, vous pouvez l'utiliser `describe-stacks` pour vérifier l'état de la création de la pile.

```
aws cloudformation describe-stacks \
      --stack-name ecs-tutorial-stack \
      --region aws-region
```

Étape 2 : vérifier la création de ressource Amazon ECS

Pour vous assurer que les ressources Amazon ECS sont créées correctement, procédez comme suit.

1. Exécutez la commande suivante pour répertorier toutes les définitions de tâches dans une Région AWS.

   ```
   aws ecs list-task-definitions
   ```

   La commande renvoie une liste des Amazon Resource Name (ARN) des définitions de tâches. L'ARN de la définition de tâche que vous avez créée à l'aide du modèle sera affiché au format suivant.

   ```
   {
       "taskDefinitionArns": [
   ```

```
     .....
        "arn:aws:ecs:aws-region:111122223333:task-definition/ecs-tutorial-stack-
task:1",
     .....
    ]
}
```

2. Exécutez la commande suivante pour répertorier tous les clusters d'une Région AWS.

```
aws ecs list-clusters
```

La commande renvoie une liste des ARN du cluster. L'ARN du cluster que vous avez créé à l'aide du modèle sera affiché au format suivant.

```
{
    "clusterArns": [
        .....
        "arn:aws:ecs:aws-region:111122223333:cluster/ecs-tutorial-stack-cluster",
        .....
    ]
}
```

3. Exécutez la commande suivante pour répertorier tous les services du cluster `ecs-tutorial-stack-cluster`.

```
aws ecs list-services \
      --cluster ecs-tutorial-stack-cluster
```

La commande renvoie une liste des ARN de services. L'ARN du service que vous avez créée à l'aide du modèle sera affiché au format suivant.

```
{
    "serviceArns": [
        "arn:aws:ecs:aws-region:111122223333:service/ecs-tutorial-stack-cluster/
ecs-tutorial-stack-service"
    ]
}
```

Vous pouvez également obtenir le nom DNS de l'Application Load Balancer créé et l'utiliser pour vérifier la création des ressources. Pour obtenir le nom DNS, exécutez la commande suivante :

Exécutez la commande suivante pour récupérer les sorties de la pile créée.

```
aws cloudformation describe-stacks \
  --stack-name ecs-tutorial-stack \
  --region aws-region \
  --query 'Stacks[0].Outputs[?OutputKey==`LoadBalancerURL`].OutputValue' \
  --output  text
```

Sortie :

```
http://ecs-tutorial-stack-alb-0123456789.aws-region.elb.amazonaws.com
```

Collez le nom DNS dans un navigateur pour afficher une page Web qui affiche un exemple d'application Amazon ECS.

Étape 3 : Nettoyer

Pour nettoyer les ressources que vous avez créées, exécutez la commande suivante.

```
aws cloudformation delete-stack \
      --stack-name ecs-stack
```

La `delete-stack` commande lance la suppression de la CloudFormation pile créée dans ce didacticiel, en supprimant toutes les ressources de la pile. Pour vérifier la suppression, vous pouvez répéter la procédure dans Étape 2 : vérifier la création de ressource Amazon ECS. La liste des ARN dans les sorties n'inclura plus de définition de tâche appelée `ecs-tutorial-stack-task` ni de cluster appelé `ecs-tutorial-stack-cluster`. L'appel `list-services` échouera.

CloudFormation exemples de modèles pour Amazon ECS

Vous pouvez créer des clusters Amazon ECS, des définitions de tâches et des services à l'aide de CloudFormation. Les rubriques suivantes incluent des modèles qui illustrent comment créer des ressources avec des configurations différentes. Vous pouvez créer ces ressources à l'aide de ces modèles à l'aide de la CloudFormation console ou du AWS CLI.

CloudFormation les modèles sont des fichiers texte au format JSON ou YAML qui décrivent les ressources que vous souhaitez fournir dans vos CloudFormation piles. Si vous ne connaissez pas le format JSON ou YAML, ou les deux, vous pouvez les utiliser AWS Infrastructure Composer pour commencer à utiliser des CloudFormation modèles. Pour plus d'informations, consultez la section

Création visuelle de modèles avec Infrastructure Composer dans le Guide de l'utilisateur AWS CloudFormation .

Les rubriques suivantes présentent des exemples de modèles pour les définitions de tâche, les clusters et les services Amazon ECS.

Rubriques

- Définitions de tâche
- Fournisseurs de capacité
- Clusters
- Services
- Rôles IAM pour Amazon ECS

Définitions de tâche

Une définition de tâche est un plan pour votre application qui décrit les paramètres et un ou plusieurs conteneurs qui constituent votre application. Vous trouverez ci-dessous des exemples CloudFormation de modèles pour les définitions de tâches Amazon ECS. Pour de plus amples informations sur les définitions de tâche Amazon ECS, consultez Définitions de tâche Amazon ECS.

Définition de tâche Fargate Linux

Vous pouvez utiliser le modèle suivant pour créer un exemple de tâche Fargate Linux.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "ECS Task Definition with parameterized values",
  "Parameters": {
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/docker/library/httpd:2.4",
      "Description": "The container image to use for the task"
    },
    "ContainerCpu": {
      "Type": "Number",
      "Default": 256,
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": [256, 512, 1024, 2048, 4096]
```

```
    },
    "ContainerMemory": {
      "Type": "Number",
      "Default": 512,
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "task-definition-cfn",
      "Description": "The name of the task definition family"
    },
    "ContainerName": {
      "Type": "String",
      "Default": "sample-fargate-app",
      "Description": "The name of the container"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "HostPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the host"
    },
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "/ecs/fargate-task-definition",
      "Description": "The CloudWatch log group for container logs"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "OperatingSystemFamily": {
```

```
      "Type": "String",
      "Default": "LINUX",
      "Description": "The operating system for the task",
      "AllowedValues": ["LINUX", "WINDOWS_SERVER_2019_FULL",
 "WINDOWS_SERVER_2019_CORE", "WINDOWS_SERVER_2022_FULL", "WINDOWS_SERVER_2022_CORE"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ContainerDefinitions": [
          {
            "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 && -
foreground\""
            ],
            "EntryPoint": [
              "sh",
              "-c"
            ],
            "Essential": true,
            "Image": {"Ref": "ContainerImage"},
            "LogConfiguration": {
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-create-group": "true",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-stream-prefix": "ecs"
              }
            },
            "Name": {"Ref": "ContainerName"},
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "HostPort": {"Ref": "HostPort"},
                "Protocol": "tcp"
```

```
              }
            ]
          }
        ],
        "Cpu": {"Ref": "ContainerCpu"},
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "Family": {"Ref": "TaskFamily"},
        "Memory": {"Ref": "ContainerMemory"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "RuntimePlatform": {
          "OperatingSystemFamily": {"Ref": "OperatingSystemFamily"}
        }
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: 'ECS Task Definition to deploy a sample app'
Parameters:
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/docker/library/httpd:2.4'
    Description: The container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    Description: The number of CPU units to reserve for the container
    AllowedValues: [256, 512, 1024, 2048, 4096]
  ContainerMemory:
    Type: Number
    Default: 512
```

```
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
  TaskFamily:
    Type: String
    Default: 'task-definition-cfn'
    Description: The name of the task definition family
  ContainerName:
    Type: String
    Default: 'sample-fargate-app'
    Description: The name of the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  HostPort:
    Type: Number
    Default: 80
    Description: The port number on the host
  ExecutionRoleArn:
    Type: String
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
    Description: The ARN of the task execution role
  LogGroup:
    Type: String
    Default: '/ecs/fargate-task-definition'
    Description: The CloudWatch log group for container logs
  NetworkMode:
    Type: String
    Default: 'awsvpc'
    Description: The Docker networking mode to use
    AllowedValues: ['awsvpc', 'bridge', 'host', 'none']
  OperatingSystemFamily:
    Type: String
    Default: 'LINUX'
    Description: The operating system for the task
    AllowedValues: ['LINUX', 'WINDOWS_SERVER_2019_FULL', 'WINDOWS_SERVER_2019_CORE',
 'WINDOWS_SERVER_2022_FULL', 'WINDOWS_SERVER_2022_CORE']
Resources:
  ECSTaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
      ContainerDefinitions:
        - 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"
          EntryPoint:
            - sh
            - '-c'
          Essential: true
          Image: !Ref ContainerImage
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-create-group: 'true'
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          Name: !Ref ContainerName
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref HostPort
              Protocol: tcp
      Cpu: !Ref ContainerCpu
      ExecutionRoleArn: !Ref ExecutionRoleArn
      Family: !Ref TaskFamily
      Memory: !Ref ContainerMemory
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      RuntimePlatform:
        OperatingSystemFamily: !Ref OperatingSystemFamily
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Définition de tâche Amazon ECS

Vous pouvez utiliser le modèle suivant pour créer une tâche qui utilise un système de fichiers Amazon EFS que vous avez créé. Pour plus d'informations sur l'utilisation des volumes Amazon EFS avec Amazon ECS, consultez la section Utilisation des volumes Amazon EFS avec Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "Create a task definition for a web server with parameterized
 values.",
  "Parameters": {
    "ExecutionRoleArn": {
      "Type": "String",
      "Default": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
      "Description": "The ARN of the task execution role"
    },
    "NetworkMode": {
      "Type": "String",
      "Default": "awsvpc",
      "Description": "The Docker networking mode to use",
      "AllowedValues": ["awsvpc", "bridge", "host", "none"]
    },
    "TaskFamily": {
      "Type": "String",
      "Default": "my-ecs-task",
      "Description": "The name of the task definition family"
    },
    "ContainerCpu": {
      "Type": "String",
      "Default": "256",
      "Description": "The number of CPU units to reserve for the container",
      "AllowedValues": ["256", "512", "1024", "2048", "4096"]
    },
    "ContainerMemory": {
      "Type": "String",
      "Default": "512",
      "Description": "The amount of memory (in MiB) to reserve for the container",
      "AllowedValues": ["512", "1024", "2048", "3072", "4096", "5120", "6144",
 "7168", "8192"]
    },
    "ContainerName": {
```

```
      "Type": "String",
      "Default": "nginx",
      "Description": "The name of the container"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "The container image to use for the task"
    },
    "ContainerPort": {
      "Type": "Number",
      "Default": 80,
      "Description": "The port number on the container"
    },
    "InitProcessEnabled": {
      "Type": "String",
      "Default": "true",
      "Description": "Whether to enable the init process inside the container",
      "AllowedValues": ["true", "false"]
    },
    "EfsVolumeName": {
      "Type": "String",
      "Default": "efs-volume",
      "Description": "The name of the EFS volume"
    },
    "EfsContainerPath": {
      "Type": "String",
      "Default": "/usr/share/nginx/html",
      "Description": "The path in the container where the EFS volume will be
mounted"
    },
    "LogGroup": {
      "Type": "String",
      "Default": "LogGroup",
      "Description": "The CloudWatch log group for container logs"
    },
    "LogStreamPrefix": {
      "Type": "String",
      "Default": "efs-task",
      "Description": "The prefix for the log stream"
    },
    "EfsFilesystemId": {
      "Type": "String",
      "Default": "fs-1234567890abcdef0",
```

```
      "Description": "The ID of the EFS filesystem"
    },
    "EfsRootDirectory": {
      "Type": "String",
      "Default": "/",
      "Description": "The root directory in the EFS filesystem"
    },
    "EfsTransitEncryption": {
      "Type": "String",
      "Default": "ENABLED",
      "Description": "Whether to enable transit encryption for EFS",
      "AllowedValues": ["ENABLED", "DISABLED"]
    }
  },
  "Resources": {
    "ECSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Ref": "ExecutionRoleArn"},
        "NetworkMode": {"Ref": "NetworkMode"},
        "RequiresCompatibilities": ["FARGATE"],
        "Family": {"Ref": "TaskFamily"},
        "Cpu": {"Ref": "ContainerCpu"},
        "Memory": {"Ref": "ContainerMemory"},
        "ContainerDefinitions": [
          {
            "Name": {"Ref": "ContainerName"},
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "ContainerPort": {"Ref": "ContainerPort"},
                "Protocol": "tcp"
              }
            ],
            "LinuxParameters": {
              "InitProcessEnabled": {"Ref": "InitProcessEnabled"}
            },
            "MountPoints": [
              {
                "SourceVolume": {"Ref": "EfsVolumeName"},
                "ContainerPath": {"Ref": "EfsContainerPath"}
              }
            ],
```

```
            "LogConfiguration": {
              "LogDriver": "awslogs",
              "Options": {
                "mode": "non-blocking",
                "max-buffer-size": "25m",
                "awslogs-group": {"Ref": "LogGroup"},
                "awslogs-region": {"Ref": "AWS::Region"},
                "awslogs-create-group": "true",
                "awslogs-stream-prefix": {"Ref": "LogStreamPrefix"}
              }
            }
          }
        ],
        "Volumes": [
          {
            "Name": {"Ref": "EfsVolumeName"},
            "EFSVolumeConfiguration": {
              "FilesystemId": {"Ref": "EfsFilesystemId"},
              "RootDirectory": {"Ref": "EfsRootDirectory"},
              "TransitEncryption": {"Ref": "EfsTransitEncryption"}
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "TaskDefinitionArn": {
      "Description": "The ARN of the created task definition",
      "Value": {"Ref": "ECSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: Create a task definition for a web server with parameterized values.
Parameters:
  ExecutionRoleArn:
    Type: String
    Default: arn:aws:iam::123456789012:role/ecsTaskExecutionRole
    Description: The ARN of the task execution role
```

```
  NetworkMode:
    Type: String
    Default: awsvpc
    Description: The Docker networking mode to use
    AllowedValues: [awsvpc, bridge, host, none]
  TaskFamily:
    Type: String
    Default: my-ecs-task
    Description: The name of the task definition family
  ContainerCpu:
    Type: String
    Default: "256"
    Description: The number of CPU units to reserve for the container
    AllowedValues: ["256", "512", "1024", "2048", "4096"]
  ContainerMemory:
    Type: String
    Default: "512"
    Description: The amount of memory (in MiB) to reserve for the container
    AllowedValues: ["512", "1024", "2048", "3072", "4096", "5120", "6144", "7168",
"8192"]
  ContainerName:
    Type: String
    Default: nginx
    Description: The name of the container
  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: The container image to use for the task
  ContainerPort:
    Type: Number
    Default: 80
    Description: The port number on the container
  InitProcessEnabled:
    Type: String
    Default: "true"
    Description: Whether to enable the init process inside the container
    AllowedValues: ["true", "false"]
  EfsVolumeName:
    Type: String
    Default: efs-volume
    Description: The name of the EFS volume
  EfsContainerPath:
    Type: String
    Default: /usr/share/nginx/html
```

```
    Description: The path in the container where the EFS volume will be mounted
  LogGroup:
    Type: String
    Default: LogGroup
    Description: The CloudWatch log group for container logs
  LogStreamPrefix:
    Type: String
    Default: efs-task
    Description: The prefix for the log stream
  EfsFilesystemId:
    Type: String
    Default: fs-1234567890abcdef0
    Description: The ID of the EFS filesystem
  EfsRootDirectory:
    Type: String
    Default: /
    Description: The root directory in the EFS filesystem
  EfsTransitEncryption:
    Type: String
    Default: ENABLED
    Description: Whether to enable transit encryption for EFS
    AllowedValues: [ENABLED, DISABLED]
Resources:
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ExecutionRoleArn: !Ref ExecutionRoleArn
      NetworkMode: !Ref NetworkMode
      RequiresCompatibilities:
        - FARGATE
      Family: !Ref TaskFamily
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      ContainerDefinitions:
        - Name: !Ref ContainerName
          Image: !Ref ContainerImage
          Essential: true
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          LinuxParameters:
            InitProcessEnabled: !Ref InitProcessEnabled
          MountPoints:
            - SourceVolume: !Ref EfsVolumeName
```

```
              ContainerPath: !Ref EfsContainerPath
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-create-group: "true"
              awslogs-stream-prefix: !Ref LogStreamPrefix
      Volumes:
        - Name: !Ref EfsVolumeName
          EFSVolumeConfiguration:
            FilesystemId: !Ref EfsFilesystemId
            RootDirectory: !Ref EfsRootDirectory
            TransitEncryption: !Ref EfsTransitEncryption
Outputs:
  TaskDefinitionArn:
    Description: The ARN of the created task definition
    Value: !Ref ECSTaskDefinition
```

Fournisseurs de capacité

Les fournisseurs de capacité sont associés à un cluster Amazon ECS et sont utilisés pour gérer la capacité de calcul pour vos charges de travail.

Création d'un fournisseur de capacité pour les instances gérées Amazon ECS

Par défaut, Amazon ECS fournit un fournisseur de capacité qui sélectionne automatiquement les types d'instances à usage général les plus optimisés en termes de coûts. Vous pouvez toutefois créer des fournisseurs de capacité personnalisés pour spécifier les exigences des instances, telles que les types d'instances, les fabricants d'UCs, les types d'accélérateurs et d'autres exigences. Vous pouvez utiliser le modèle suivant pour créer un fournisseur de capacité pour les instances gérées Amazon ECS qui répond aux exigences de mémoire et d'UC spécifiées.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "MyCapacityProvider": {
```

```
            "Type": "AWS::ECS::CapacityProvider",
            "Properties": {
                "ManagedInstancesProvider": {
                    "InfrastructureRoleArn": "arn:aws:iam::123456789012:role/
ECSInfrastructureRole",
                    "InstanceLaunchTemplate": {
                        "Ec2InstanceProfileArn":
 "arn:aws:iam::123456789012:instance-profile/ecsInstanceProfile",
                        "NetworkConfiguration": null,
                        "Subnets": [
                            "subnet-12345678"
                        ],
                        "SecurityGroups": [
                            "sg-87654321"
                        ]
                    },
                    "StorageConfiguration": {
                        "StorageSizeGiB": 30
                    },
                    "InstanceRequirements": {
                        "VCpuCount": {
                            "Min": 1,
                            "Max": 4
                        },
                        "MemoryMiB": {
                            "Min": 2048,
                            "Max": 8192
                        }
                    }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCapacityProvider:
    Type: AWS::ECS::CapacityProvider
    Properties:
      ManagedInstancesProvider:
```

```
        InfrastructureRoleArn: "arn:aws:iam::123456789012:role/
ECSInfrastructureRole"
        InstanceLaunchTemplate:
          Ec2InstanceProfileArn: "arn:aws:iam::123456789012:instance-profile/
ecsInstanceProfile"
          NetworkConfiguration:
          Subnets:
            - "subnet-12345678"
          SecurityGroups:
            - "sg-87654321"
        StorageConfiguration:
          StorageSizeGiB: 30
        InstanceRequirements:
          VCpuCount:
            Min: 1
            Max: 4
          MemoryMiB:
            Min: 2048
            Max: 8192
```

Clusters

Un cluster Amazon ECS est un regroupement logique de tâches ou de services. Vous pouvez utiliser les modèles suivants pour créer des clusters avec des configurations différentes. Pour plus d'informations sur les clusters Amazon ECS, consultez la section Clusters Amazon ECS.

Création d'un cluster vide avec les paramètres par défaut

Vous pouvez utiliser le modèle suivant pour créer un cluster vide avec les paramètres par défaut.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "MyEmptyCluster"
            }
        }
    }
```

```
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  ECSCluster:
    Type: 'AWS::ECS::Cluster'
    Properties:
      ClusterName: MyEmptyCluster
```

Création d'un cluster vide avec chiffrement du stockage géré et fonctionnalité Container Insights améliorée

Vous pouvez utiliser le modèle suivant pour créer un cluster avec un stockage géré au niveau du cluster et activé Container Insights amélioré. Cluster-level le chiffrement s'applique aux volumes de données gérés par Amazon ECS tels que les volumes Amazon EBS. Pour plus d'informations sur le chiffrement Amazon EFS, consultez la section Chiffrement des données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS. Pour plus d'informations sur l'utilisation de Container Insights avec observabilité améliorée, consultez la section Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
   "Resources": {
        "Cluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": "EncryptedEnhancedCluster",
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enhanced"
                    }
                ],
                "Configuration": {
                    "ManagedStorageConfiguration": {
                        "KmsKeyId": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

```
                    }
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  Cluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: EncryptedEnhancedCluster
      ClusterSettings:
        - Name: containerInsights
          Value: enhanced
      Configuration:
        ManagedStorageConfiguration:
          KmsKeyId: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
```

Créez un cluster avec l'Amazon AL2023 ECS-Optimized-AMI

Vous pouvez utiliser le modèle suivant pour créer un cluster qui utilise un fournisseur de capacité qui lance des instances AL2023 sur Amazon EC2.

> **Important**
>
> Pour obtenir les derniers ID d'AMI, consultez Amazon ECS-optimized AMI dans le guide du développeur Amazon Elastic Container Service.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "EC2 ECS cluster that starts out empty, with no EC2 instances
 yet. An ECS capacity provider automatically launches more EC2 instances as required
 on the fly when you request ECS to launch services or standalone tasks.",
    "Parameters": {
```

```
        "InstanceType": {
            "Type": "String",
            "Description": "EC2 instance type",
            "Default": "t2.medium",
            "AllowedValues": [
                "t1.micro",
                "t2.2xlarge",
                "t2.large",
                "t2.medium",
                "t2.micro",
                "t2.nano",
                "t2.small",
                "t2.xlarge",
                "t3.2xlarge",
                "t3.large",
                "t3.medium",
                "t3.micro",
                "t3.nano",
                "t3.small",
                "t3.xlarge"
            ]
        },
        "DesiredCapacity": {
            "Type": "Number",
            "Default": "0",
            "Description": "Number of EC2 instances to launch in your ECS cluster."
        },
        "MaxSize": {
            "Type": "Number",
            "Default": "100",
            "Description": "Maximum number of EC2 instances that can be launched in
 your ECS cluster."
        },
        "ECSAMI": {
            "Description": "The Amazon Machine Image ID used for the cluster",
            "Type": "AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>",
            "Default": "/aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id"
        },
        "VpcId": {
            "Type": "AWS::EC2::VPC::Id",
            "Description": "VPC ID where the ECS cluster is launched",
            "Default": "vpc-1234567890abcdef0"
        },
```

```
        "SubnetIds": {
            "Type": "List<AWS::EC2::Subnet::Id>",
            "Description": "List of subnet IDs where the EC2 instances will be
launched",
            "Default": "subnet-021345abcdef67890"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enabled"
                    }
                ]
            }
        },
        "ECSAutoScalingGroup": {
            "Type": "AWS::AutoScaling::AutoScalingGroup",
            "DependsOn": [
                "ECSCluster",
                "EC2Role"
            ],
            "Properties": {
                "VPCZoneIdentifier": {
                    "Ref": "SubnetIds"
                },
                "LaunchTemplate": {
                    "LaunchTemplateId": {
                        "Ref": "ContainerInstances"
                    },
                    "Version": {
                        "Fn::GetAtt": [
                            "ContainerInstances",
                            "LatestVersionNumber"
                        ]
                    }
                },
                "MinSize": 0,
                "MaxSize": {
                    "Ref": "MaxSize"
                },
```

```
                "DesiredCapacity": {
                    "Ref": "DesiredCapacity"
                },
                "NewInstancesProtectedFromScaleIn": true
            },
            "UpdatePolicy": {
                "AutoScalingReplacingUpdate": {
                    "WillReplace": "true"
                }
            }
        },
        "ContainerInstances": {
            "Type": "AWS::EC2::LaunchTemplate",
            "Properties": {
                "LaunchTemplateName": "asg-launch-template-2",
                "LaunchTemplateData": {
                    "ImageId": {
                        "Ref": "ECSAMI"
                    },
                    "InstanceType": {
                        "Ref": "InstanceType"
                    },
                    "IamInstanceProfile": {
                        "Name": {
                            "Ref": "EC2InstanceProfile"
                        }
                    },
                    "SecurityGroupIds": [
                        {
                            "Ref": "ContainerHostSecurityGroup"
                        }
                    ],
                    "UserData": {
                        "Fn::Base64": {
                            "Fn::Sub": "#!/bin/bash -xe\n echo ECS_CLUSTER=
${ECSCluster} >> /etc/ecs/ecs.config\n yum install -y aws-cfn-bootstrap\n /opt/aws/
bin/cfn-init -v --stack ${AWS::StackId} --resource ContainerInstances --configsets
 full_install --region ${AWS::Region} &\n"
                        }
                    },
                    "MetadataOptions": {
                        "HttpEndpoint": "enabled",
                        "HttpTokens": "required"
                    }
```

```
                }
            }
        },
        "EC2InstanceProfile": {
            "Type": "AWS::IAM::InstanceProfile",
            "Properties": {
                "Path": "/",
                "Roles": [
                    {
                        "Ref": "EC2Role"
                    }
                ]
            }
        },
        "CapacityProvider": {
            "Type": "AWS::ECS::CapacityProvider",
            "Properties": {
                "AutoScalingGroupProvider": {
                    "AutoScalingGroupArn": {
                        "Ref": "ECSAutoScalingGroup"
                    },
                    "ManagedScaling": {
                        "InstanceWarmupPeriod": 60,
                        "MinimumScalingStepSize": 1,
                        "MaximumScalingStepSize": 100,
                        "Status": "ENABLED",
                        "TargetCapacity": 100
                    },
                    "ManagedTerminationProtection": "ENABLED"
                }
            }
        },
        "CapacityProviderAssociation": {
            "Type": "AWS::ECS::ClusterCapacityProviderAssociations",
            "Properties": {
                "CapacityProviders": [
                    {
                        "Ref": "CapacityProvider"
                    }
                ],
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "DefaultCapacityProviderStrategy": [
```

```
                    {
                        "Base": 0,
                        "CapacityProvider": {
                            "Ref": "CapacityProvider"
                        },
                        "Weight": 1
                    }
                ]
            }
        },
        "ContainerHostSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupDescription": "Access to the EC2 hosts that run containers",
                "VpcId": {
                    "Ref": "VpcId"
                }
            }
        },
        "EC2Role": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ec2.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ]
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role",
                    "arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore"
                ]
            }
```

```
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "AssumeRolePolicyDocument": {
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": [
                                    "ecs-tasks.amazonaws.com"
                                ]
                            },
                            "Action": [
                                "sts:AssumeRole"
                            ],
                            "Condition": {
                                "ArnLike": {
                                    "aws:SourceArn": {
                                        "Fn::Sub": "arn:${AWS::Partition}:ecs:
${AWS::Region}:${AWS::AccountId}:*"
                                    }
                                },
                                "StringEquals": {
                                    "aws:SourceAccount": {
                                        "Fn::Sub": "${AWS::AccountId}"
                                    }
                                }
                            }
                        }
                    ]
                },
                "Path": "/",
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The ECS cluster into which to launch resources",
            "Value": "ECSCluster"
```

```
        },
        "ECSTaskExecutionRole": {
            "Description": "The role used to start up a task",
            "Value": "ECSTaskExecutionRole"
        },
        "CapacityProvider": {
            "Description": "The cluster capacity provider that the service should
 use to request capacity when it wants to start up a task",
            "Value": "CapacityProvider"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: EC2 ECS cluster that starts out empty, with no EC2 instances yet. An
 ECS capacity provider automatically launches more EC2 instances as required on the
 fly when you request ECS to launch services or standalone tasks.
Parameters:
  InstanceType:
    Type: String
    Description: EC2 instance type
    Default: t2.medium
    AllowedValues:
      - t1.micro
      - t2.2xlarge
      - t2.large
      - t2.medium
      - t2.micro
      - t2.nano
      - t2.small
      - t2.xlarge
      - t3.2xlarge
      - t3.large
      - t3.medium
      - t3.micro
      - t3.nano
      - t3.small
      - t3.xlarge
  DesiredCapacity:
    Type: Number
    Default: '0'
```

```
    Description: Number of EC2 instances to launch in your ECS cluster.
  MaxSize:
    Type: Number
    Default: '100'
    Description: Maximum number of EC2 instances that can be launched in your ECS
 cluster.
  ECSAMI:
    Description: The Amazon Machine Image ID used for the cluster
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2023/recommended/image_id
  VpcId:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID where the ECS cluster is launched
    Default: vpc-1234567890abcdef0
  SubnetIds:
    Type: List<AWS::EC2::Subnet::Id>
    Description: List of subnet IDs where the EC2 instances will be launched
    Default: subnet-021345abcdef67890
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterSettings:
        - Name: containerInsights
          Value: enabled
  ECSAutoScalingGroup:
    Type: AWS::AutoScaling::AutoScalingGroup
    DependsOn:
      - ECSCluster
      - EC2Role
    Properties:
      VPCZoneIdentifier: !Ref SubnetIds
      LaunchTemplate:
        LaunchTemplateId: !Ref ContainerInstances
        Version: !GetAtt ContainerInstances.LatestVersionNumber
      MinSize: 0
      MaxSize: !Ref MaxSize
      DesiredCapacity: !Ref DesiredCapacity
      NewInstancesProtectedFromScaleIn: true
    UpdatePolicy:
      AutoScalingReplacingUpdate:
        WillReplace: 'true'
  ContainerInstances:
    Type: AWS::EC2::LaunchTemplate
```

```
    Properties:
      LaunchTemplateName: asg-launch-template-2
      LaunchTemplateData:
        ImageId: !Ref ECSAMI
        InstanceType: !Ref InstanceType
        IamInstanceProfile:
          Name: !Ref EC2InstanceProfile
        SecurityGroupIds:
          - !Ref ContainerHostSecurityGroup
        UserData: !Base64
          Fn::Sub: |
            #!/bin/bash -xe
            echo ECS_CLUSTER=${ECSCluster} >> /etc/ecs/ecs.config
            yum install -y aws-cfn-bootstrap
            /opt/aws/bin/cfn-init -v --stack ${AWS::StackId} --resource
ContainerInstances --configsets full_install --region ${AWS::Region} &
        MetadataOptions:
          HttpEndpoint: enabled
          HttpTokens: required
  EC2InstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      Path: /
      Roles:
        - !Ref EC2Role
  CapacityProvider:
    Type: AWS::ECS::CapacityProvider
    Properties:
      AutoScalingGroupProvider:
        AutoScalingGroupArn: !Ref ECSAutoScalingGroup
        ManagedScaling:
          InstanceWarmupPeriod: 60
          MinimumScalingStepSize: 1
          MaximumScalingStepSize: 100
          Status: ENABLED
          TargetCapacity: 100
        ManagedTerminationProtection: ENABLED
  CapacityProviderAssociation:
    Type: AWS::ECS::ClusterCapacityProviderAssociations
    Properties:
      CapacityProviders:
        - !Ref CapacityProvider
      Cluster: !Ref ECSCluster
      DefaultCapacityProviderStrategy:
```

```
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
  ContainerHostSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupDescription: Access to the EC2 hosts that run containers
      VpcId: !Ref VpcId
  EC2Role:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ec2.amazonaws.com
            Action:
              - sts:AssumeRole
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
        - arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service:
                - ecs-tasks.amazonaws.com
            Action:
              - sts:AssumeRole
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:${AWS::Partition}:ecs:${AWS::Region}:
${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Sub ${AWS::AccountId}
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
Outputs:
```

```
  ClusterName:
    Description: The ECS cluster into which to launch resources
    Value: ECSCluster
  ECSTaskExecutionRole:
    Description: The role used to start up a task
    Value: ECSTaskExecutionRole
  CapacityProvider:
    Description: The cluster capacity provider that the service should use to
request capacity when it wants to start up a task
    Value: CapacityProvider
```

Services

Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de vos tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service. Les modèles suivants peuvent être utilisés pour déployer des services. Pour plus d'informations sur les limites de service par défaut d'Amazon ECS, consultez la section Services Amazon ECS.

Déploiement d'une application Web à charge équilibrée

Le modèle suivant crée un service Amazon ECS avec deux tâches qui s'exécutent sur Fargate. Les tâches possèdent chacune un conteneur NGINX. Le modèle crée également un Application Load Balancer qui distribue le trafic des applications et une politique Application Auto Scaling qui adapte l'application en fonction de l'utilisation de l'UC. Le modèle crée également les ressources réseau nécessaires au déploiement de l'application, les ressources de journalisation pour les journaux de conteneurs et un rôle IAM d'exécution de tâches Amazon ECS. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS. Pour plus d'informations sur l'autoscaling, consultez la section Mise à l'échelle automatique de votre service Amazon ECS.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "[AWSDocs] ECS: load-balanced-web-application",
    "Parameters": {
        "VpcCidr": {
```

```
            "Type": "String",
            "Default": "10.0.0.0/16",
            "Description": "CIDR block for the VPC"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest",
            "Description": "Container image to use in task definition"
        },
        "PublicSubnet1Cidr": {
            "Type": "String",
            "Default": "10.0.1.0/24",
            "Description": "CIDR block for public subnet 1"
        },
        "PublicSubnet2Cidr": {
            "Type": "String",
            "Default": "10.0.2.0/24",
            "Description": "CIDR block for public subnet 2"
        },
        "PrivateSubnet1Cidr": {
            "Type": "String",
            "Default": "10.0.3.0/24",
            "Description": "CIDR block for private subnet 1"
        },
        "PrivateSubnet2Cidr": {
            "Type": "String",
            "Default": "10.0.4.0/24",
            "Description": "CIDR block for private subnet 2"
        },
        "ServiceName": {
            "Type": "String",
            "Default": "tutorial-app",
            "Description": "Name of the ECS service"
        },
        "ContainerPort": {
            "Type": "Number",
            "Default": 80,
            "Description": "Port on which the container listens"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 2,
            "Description": "Desired number of tasks"
        },
```

```
        "MinCapacity": {
            "Type": "Number",
            "Default": 1,
            "Description": "Minimum number of tasks for auto scaling"
        },
        "MaxCapacity": {
            "Type": "Number",
            "Default": 10,
            "Description": "Maximum number of tasks for auto scaling"
        }
    },
    "Resources": {
        "VPC": {
            "Type": "AWS::EC2::VPC",
            "Properties": {
                "CidrBlock": {
                    "Ref": "VpcCidr"
                },
                "EnableDnsHostnames": true,
                "EnableDnsSupport": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-vpc"
                        }
                    }
                ]
            }
        },
        "InternetGateway": {
            "Type": "AWS::EC2::InternetGateway",
            "Properties": {
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-igw"
                        }
                    }
                ]
            }
        },
        "InternetGatewayAttachment": {
```

```
            "Type": "AWS::EC2::VPCGatewayAttachment",
            "Properties": {
                "InternetGatewayId": {
                    "Ref": "InternetGateway"
                },
                "VpcId": {
                    "Ref": "VPC"
                }
            }
        },
        "PublicSubnet1": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        0,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet1Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-public-subnet-1"
                        }
                    }
                ]
            }
        },
        "PublicSubnet2": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
```

```
                "AvailabilityZone": {
                    "Fn::Select": [
                        1,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PublicSubnet2Cidr"
                },
                "MapPublicIpOnLaunch": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-public-subnet-2"
                        }
                    }
                ]
            }
        },
        "PrivateSubnet1": {
            "Type": "AWS::EC2::Subnet",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "AvailabilityZone": {
                    "Fn::Select": [
                        0,
                        {
                            "Fn::GetAZs": ""
                        }
                    ]
                },
                "CidrBlock": {
                    "Ref": "PrivateSubnet1Cidr"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-private-subnet-1"
```

```
                    }
                }
            ]
        }
    },
    "PrivateSubnet2": {
        "Type": "AWS::EC2::Subnet",
        "Properties": {
            "VpcId": {
                "Ref": "VPC"
            },
            "AvailabilityZone": {
                "Fn::Select": [
                    1,
                    {
                        "Fn::GetAZs": ""
                    }
                ]
            },
            "CidrBlock": {
                "Ref": "PrivateSubnet2Cidr"
            },
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-private-subnet-2"
                    }
                }
            ]
        }
    },
    "NatGateway1EIP": {
        "Type": "AWS::EC2::EIP",
        "DependsOn": "InternetGatewayAttachment",
        "Properties": {
            "Domain": "vpc",
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-nat-eip-1"
                    }
                }
```

```
                ]
            }
        },
        "NatGateway2EIP": {
            "Type": "AWS::EC2::EIP",
            "DependsOn": "InternetGatewayAttachment",
            "Properties": {
                "Domain": "vpc",
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-nat-eip-2"
                        }
                    }
                ]
            }
        },
        "NatGateway1": {
            "Type": "AWS::EC2::NatGateway",
            "Properties": {
                "AllocationId": {
                    "Fn::GetAtt": [
                        "NatGateway1EIP",
                        "AllocationId"
                    ]
                },
                "SubnetId": {
                    "Ref": "PublicSubnet1"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-nat-1"
                        }
                    }
                ]
            }
        },
        "NatGateway2": {
            "Type": "AWS::EC2::NatGateway",
            "Properties": {
                "AllocationId": {
```

```
                    "Fn::GetAtt": [
                        "NatGateway2EIP",
                        "AllocationId"
                    ]
                },
                "SubnetId": {
                    "Ref": "PublicSubnet2"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-nat-2"
                        }
                    }
                ]
            }
        },
        "PublicRouteTable": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-public-routes"
                        }
                    }
                ]
            }
        },
        "DefaultPublicRoute": {
            "Type": "AWS::EC2::Route",
            "DependsOn": "InternetGatewayAttachment",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "GatewayId": {
                    "Ref": "InternetGateway"
```

```
                }
            }
        },
        "PublicSubnet1RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet1"
                }
            }
        },
        "PublicSubnet2RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PublicRouteTable"
                },
                "SubnetId": {
                    "Ref": "PublicSubnet2"
                }
            }
        },
        "PrivateRouteTable1": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-private-routes-1"
                        }
                    }
                ]
            }
        },
        "DefaultPrivateRoute1": {
            "Type": "AWS::EC2::Route",
            "Properties": {
```

```
                "RouteTableId": {
                    "Ref": "PrivateRouteTable1"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "NatGatewayId": {
                    "Ref": "NatGateway1"
                }
            }
        },
        "PrivateSubnet1RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PrivateRouteTable1"
                },
                "SubnetId": {
                    "Ref": "PrivateSubnet1"
                }
            }
        },
        "PrivateRouteTable2": {
            "Type": "AWS::EC2::RouteTable",
            "Properties": {
                "VpcId": {
                    "Ref": "VPC"
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-private-routes-2"
                        }
                    }
                ]
            }
        },
        "DefaultPrivateRoute2": {
            "Type": "AWS::EC2::Route",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PrivateRouteTable2"
                },
                "DestinationCidrBlock": "0.0.0.0/0",
                "NatGatewayId": {
```

```
                    "Ref": "NatGateway2"
                }
            }
        },
        "PrivateSubnet2RouteTableAssociation": {
            "Type": "AWS::EC2::SubnetRouteTableAssociation",
            "Properties": {
                "RouteTableId": {
                    "Ref": "PrivateRouteTable2"
                },
                "SubnetId": {
                    "Ref": "PrivateSubnet2"
                }
            }
        },
        "ALBSecurityGroup": {
            "Type": "AWS::EC2::SecurityGroup",
            "Properties": {
                "GroupName": {
                    "Fn::Sub": "${AWS::StackName}-alb-sg"
                },
                "GroupDescription": "Security group for Application Load Balancer",
                "VpcId": {
                    "Ref": "VPC"
                },
                "SecurityGroupIngress": [
                    {
                        "IpProtocol": "tcp",
                        "FromPort": 80,
                        "ToPort": 80,
                        "CidrIp": "0.0.0.0/0",
                        "Description": "Allow HTTP traffic from internet"
                    }
                ],
                "SecurityGroupEgress": [
                    {
                        "IpProtocol": -1,
                        "CidrIp": "0.0.0.0/0",
                        "Description": "Allow all outbound traffic"
                    }
                ],
                "Tags": [
                    {
                        "Key": "Name",
```

```
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-alb-sg"
                    }
                }
            ]
        }
    },
    "ECSSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupName": {
                "Fn::Sub": "${AWS::StackName}-ecs-sg"
            },
            "GroupDescription": "Security group for ECS tasks",
            "VpcId": {
                "Ref": "VPC"
            },
            "SecurityGroupIngress": [
                {
                    "IpProtocol": "tcp",
                    "FromPort": {
                        "Ref": "ContainerPort"
                    },
                    "ToPort": {
                        "Ref": "ContainerPort"
                    },
                    "SourceSecurityGroupId": {
                        "Ref": "ALBSecurityGroup"
                    },
                    "Description": "Allow traffic from ALB"
                }
            ],
            "SecurityGroupEgress": [
                {
                    "IpProtocol": -1,
                    "CidrIp": "0.0.0.0/0",
                    "Description": "Allow all outbound traffic"
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-ecs-sg"
```

```
                    }
                }
            ]
        }
    },
    "ApplicationLoadBalancer": {
        "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
        "Properties": {
            "Name": {
                "Fn::Sub": "${AWS::StackName}-alb"
            },
            "Scheme": "internet-facing",
            "Type": "application",
            "Subnets": [
                {
                    "Ref": "PublicSubnet1"
                },
                {
                    "Ref": "PublicSubnet2"
                }
            ],
            "SecurityGroups": [
                {
                    "Ref": "ALBSecurityGroup"
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-alb"
                    }
                }
            ]
        }
    },
    "ALBTargetGroup": {
        "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
        "Properties": {
            "Name": {
                "Fn::Sub": "${AWS::StackName}-tg"
            },
            "Port": {
                "Ref": "ContainerPort"
```

```
                },
                "Protocol": "HTTP",
                "VpcId": {
                    "Ref": "VPC"
                },
                "TargetType": "ip",
                "HealthCheckIntervalSeconds": 30,
                "HealthCheckPath": "/",
                "HealthCheckProtocol": "HTTP",
                "HealthCheckTimeoutSeconds": 5,
                "HealthyThresholdCount": 2,
                "UnhealthyThresholdCount": 5,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-tg"
                        }
                    }
                ]
            }
        },
        "ALBListener": {
            "Type": "AWS::ElasticLoadBalancingV2::Listener",
            "Properties": {
                "DefaultActions": [
                    {
                        "Type": "forward",
                        "TargetGroupArn": {
                            "Ref": "ALBTargetGroup"
                        }
                    }
                ],
                "LoadBalancerArn": {
                    "Ref": "ApplicationLoadBalancer"
                },
                "Port": 80,
                "Protocol": "HTTP"
            }
        },
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {
```

```
                    "Fn::Sub": "${AWS::StackName}-cluster"
                },
                "CapacityProviders": [
                    "FARGATE",
                    "FARGATE_SPOT"
                ],
                "DefaultCapacityProviderStrategy": [
                    {
                        "CapacityProvider": "FARGATE",
                        "Weight": 1
                    },
                    {
                        "CapacityProvider": "FARGATE_SPOT",
                        "Weight": 4
                    }
                ],
                "ClusterSettings": [
                    {
                        "Name": "containerInsights",
                        "Value": "enabled"
                    }
                ],
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-cluster"
                        }
                    }
                ]
            }
        },
        "ECSTaskExecutionRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "RoleName": {
                    "Fn::Sub": "${AWS::StackName}-task-execution-role"
                },
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
```

```
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "ManagedPolicyArns": [
                    "arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy"
                ],
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-task-execution-role"
                        }
                    }
                ]
            }
        },
        "ECSTaskRole": {
            "Type": "AWS::IAM::Role",
            "Properties": {
                "RoleName": {
                    "Fn::Sub": "${AWS::StackName}-task-role"
                },
                "AssumeRolePolicyDocument": {
                    "Version": "2012-10-17",
                    "Statement": [
                        {
                            "Effect": "Allow",
                            "Principal": {
                                "Service": "ecs-tasks.amazonaws.com"
                            },
                            "Action": "sts:AssumeRole"
                        }
                    ]
                },
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-task-role"
                        }
```

```
                }
            ]
        }
    },
    "LogGroup": {
        "Type": "AWS::Logs::LogGroup",
        "Properties": {
            "LogGroupName": {
                "Fn::Sub": "/ecs/${AWS::StackName}"
            },
            "RetentionInDays": 7
        }
    },
    "TaskDefinition": {
        "Type": "AWS::ECS::TaskDefinition",
        "Properties": {
            "Family": {
                "Fn::Sub": "${AWS::StackName}-task"
            },
            "Cpu": "256",
            "Memory": "512",
            "NetworkMode": "awsvpc",
            "RequiresCompatibilities": [
                "FARGATE"
            ],
            "ExecutionRoleArn": {
                "Fn::GetAtt": [
                    "ECSTaskExecutionRole",
                    "Arn"
                ]
            },
            "TaskRoleArn": {
                "Fn::GetAtt": [
                    "ECSTaskRole",
                    "Arn"
                ]
            },
            "ContainerDefinitions": [
                {
                    "Name": {
                        "Ref": "ServiceName"
                    },
                    "Image": {
                        "Ref": "ContainerImage"
```

```
                    },
                    "PortMappings": [
                        {
                            "ContainerPort": {
                                "Ref": "ContainerPort"
                            },
                            "Protocol": "tcp"
                        }
                    ],
                    "Essential": true,
                    "LogConfiguration": {
                        "LogDriver": "awslogs",
                        "Options": {
                            "awslogs-group": {
                                "Ref": "LogGroup"
                            },
                            "awslogs-region": {
                                "Ref": "AWS::Region"
                            },
                            "awslogs-stream-prefix": "ecs"
                        }
                    },
                    "HealthCheck": {
                        "Command": [
                            "CMD-SHELL",
                            "curl -f http://localhost/ || exit 1"
                        ],
                        "Interval": 30,
                        "Timeout": 5,
                        "Retries": 3,
                        "StartPeriod": 60
                    }
                }
            ],
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {
                        "Fn::Sub": "${AWS::StackName}-task"
                    }
                }
            ]
        }
    },
```

```
        "ECSService": {
            "Type": "AWS::ECS::Service",
            "DependsOn": "ALBListener",
            "Properties": {
                "ServiceName": {
                    "Fn::Sub": "${AWS::StackName}-service"
                },
                "Cluster": {
                    "Ref": "ECSCluster"
                },
                "TaskDefinition": {
                    "Ref": "TaskDefinition"
                },
                "DesiredCount": {
                    "Ref": "DesiredCount"
                },
                "LaunchType": "FARGATE",
                "PlatformVersion": "LATEST",
                "NetworkConfiguration": {
                    "AwsvpcConfiguration": {
                        "AssignPublicIp": "DISABLED",
                        "SecurityGroups": [
                            {
                                "Ref": "ECSSecurityGroup"
                            }
                        ],
                        "Subnets": [
                            {
                                "Ref": "PrivateSubnet1"
                            },
                            {
                                "Ref": "PrivateSubnet2"
                            }
                        ]
                    }
                },
                "LoadBalancers": [
                    {
                        "ContainerName": {
                            "Ref": "ServiceName"
                        },
                        "ContainerPort": {
                            "Ref": "ContainerPort"
                        },
```

```
                        "TargetGroupArn": {
                            "Ref": "ALBTargetGroup"
                        }
                    }
                ],
                "DeploymentConfiguration": {
                    "MaximumPercent": 200,
                    "MinimumHealthyPercent": 50,
                    "DeploymentCircuitBreaker": {
                        "Enable": true,
                        "Rollback": true
                    }
                },
                "EnableExecuteCommand": true,
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {
                            "Fn::Sub": "${AWS::StackName}-service"
                        }
                    }
                ]
            }
        },
        "ServiceScalingTarget": {
            "Type": "AWS::ApplicationAutoScaling::ScalableTarget",
            "Properties": {
                "MaxCapacity": {
                    "Ref": "MaxCapacity"
                },
                "MinCapacity": {
                    "Ref": "MinCapacity"
                },
                "ResourceId": {
                    "Fn::Sub": "service/${ECSCluster}/${ECSService.Name}"
                },
                "RoleARN": {
                    "Fn::Sub": "arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService"
                },
                "ScalableDimension": "ecs:service:DesiredCount",
                "ServiceNamespace": "ecs"
            }
```

```
        },
        "ServiceScalingPolicy": {
            "Type": "AWS::ApplicationAutoScaling::ScalingPolicy",
            "Properties": {
                "PolicyName": {
                    "Fn::Sub": "${AWS::StackName}-cpu-scaling-policy"
                },
                "PolicyType": "TargetTrackingScaling",
                "ScalingTargetId": {
                    "Ref": "ServiceScalingTarget"
                },
                "TargetTrackingScalingPolicyConfiguration": {
                    "PredefinedMetricSpecification": {
                        "PredefinedMetricType": "ECSServiceAverageCPUUtilization"
                    },
                    "TargetValue": 70,
                    "ScaleOutCooldown": 300,
                    "ScaleInCooldown": 300
                }
            }
        }
    },
    "Outputs": {
        "VPCId": {
            "Description": "VPC ID",
            "Value": {
                "Ref": "VPC"
            },
            "Export": {
                "Name": {
                    "Fn::Sub": "${AWS::StackName}-VPC-ID"
                }
            }
        },
        "LoadBalancerURL": {
            "Description": "URL of the Application Load Balancer",
            "Value": {
                "Fn::Sub": "http://${ApplicationLoadBalancer.DNSName}"
            },
            "Export": {
                "Name": {
                    "Fn::Sub": "${AWS::StackName}-ALB-URL"
                }
            }
```

```
        },
        "ECSClusterName": {
            "Description": "Name of the ECS Cluster",
            "Value": {
                "Ref": "ECSCluster"
            },
            "Export": {
                "Name": {
                    "Fn::Sub": "${AWS::StackName}-ECS-Cluster"
                }
            }
        },
        "ECSServiceName": {
            "Description": "Name of the ECS Service",
            "Value": {
                "Fn::GetAtt": [
                    "ECSService",
                    "Name"
                ]
            },
            "Export": {
                "Name": {
                    "Fn::Sub": "${AWS::StackName}-ECS-Service"
                }
            }
        },
        "PrivateSubnet1": {
            "Description": "Private Subnet 1 ID",
            "Value": {
                "Ref": "PrivateSubnet1"
            },
            "Export": {
                "Name": {
                    "Fn::Sub": "${AWS::StackName}-Private-Subnet-1"
                }
            }
        },
        "PrivateSubnet2": {
            "Description": "Private Subnet 2 ID",
            "Value": {
                "Ref": "PrivateSubnet2"
            },
            "Export": {
                "Name": {
```

```
                    "Fn::Sub": "${AWS::StackName}-Private-Subnet-2"
                }
            }
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: '[AWSDocs] ECS: load-balanced-web-application'

Parameters:
  VpcCidr:
    Type: String
    Default: '10.0.0.0/16'
    Description: CIDR block for the VPC
  ContainerImage:
    Type: String
    Default: 'public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest'
    Description: Container image to use in task definition

  PublicSubnet1Cidr:
    Type: String
    Default: '10.0.1.0/24'
    Description: CIDR block for public subnet 1

  PublicSubnet2Cidr:
    Type: String
    Default: '10.0.2.0/24'
    Description: CIDR block for public subnet 2

  PrivateSubnet1Cidr:
    Type: String
    Default: '10.0.3.0/24'
    Description: CIDR block for private subnet 1

  PrivateSubnet2Cidr:
    Type: String
    Default: '10.0.4.0/24'
    Description: CIDR block for private subnet 2

  ServiceName:
```

```
    Type: String
    Default: 'tutorial-app'
    Description: Name of the ECS service

  ContainerPort:
    Type: Number
    Default: 80
    Description: Port on which the container listens

  DesiredCount:
    Type: Number
    Default: 2
    Description: Desired number of tasks

  MinCapacity:
    Type: Number
    Default: 1
    Description: Minimum number of tasks for auto scaling

  MaxCapacity:
    Type: Number
    Default: 10
    Description: Maximum number of tasks for auto scaling

Resources:
  # VPC and Networking
  VPC:
    Type: AWS::EC2::VPC
    Properties:
      CidrBlock: !Ref VpcCidr
      EnableDnsHostnames: true
      EnableDnsSupport: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-vpc'

  # Internet Gateway
  InternetGateway:
    Type: AWS::EC2::InternetGateway
    Properties:
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-igw'
```

```
  InternetGatewayAttachment:
    Type: AWS::EC2::VPCGatewayAttachment
    Properties:
      InternetGatewayId: !Ref InternetGateway
      VpcId: !Ref VPC

  # Public Subnets for ALB
  PublicSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PublicSubnet1Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-1'

  PublicSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PublicSubnet2Cidr
      MapPublicIpOnLaunch: true
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-subnet-2'

  # Private Subnets for ECS Tasks
  PrivateSubnet1:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
      AvailabilityZone: !Select [0, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet1Cidr
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-subnet-1'

  PrivateSubnet2:
    Type: AWS::EC2::Subnet
    Properties:
      VpcId: !Ref VPC
```

```
      AvailabilityZone: !Select [1, !GetAZs '']
      CidrBlock: !Ref PrivateSubnet2Cidr
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-subnet-2'

  # NAT Gateways for private subnet internet access
  NatGateway1EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-1'

  NatGateway2EIP:
    Type: AWS::EC2::EIP
    DependsOn: InternetGatewayAttachment
    Properties:
      Domain: vpc
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-eip-2'

  NatGateway1:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway1EIP.AllocationId
      SubnetId: !Ref PublicSubnet1
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-1'

  NatGateway2:
    Type: AWS::EC2::NatGateway
    Properties:
      AllocationId: !GetAtt NatGateway2EIP.AllocationId
      SubnetId: !Ref PublicSubnet2
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-nat-2'

  # Route Tables
```

```
  PublicRouteTable:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-public-routes'

  DefaultPublicRoute:
    Type: AWS::EC2::Route
    DependsOn: InternetGatewayAttachment
    Properties:
      RouteTableId: !Ref PublicRouteTable
      DestinationCidrBlock: 0.0.0.0/0
      GatewayId: !Ref InternetGateway

  PublicSubnet1RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet1

  PublicSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PublicRouteTable
      SubnetId: !Ref PublicSubnet2

  PrivateRouteTable1:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-1'

  DefaultPrivateRoute1:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway1

  PrivateSubnet1RouteTableAssociation:
```

```
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable1
      SubnetId: !Ref PrivateSubnet1

  PrivateRouteTable2:
    Type: AWS::EC2::RouteTable
    Properties:
      VpcId: !Ref VPC
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-private-routes-2'

  DefaultPrivateRoute2:
    Type: AWS::EC2::Route
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      DestinationCidrBlock: 0.0.0.0/0
      NatGatewayId: !Ref NatGateway2

  PrivateSubnet2RouteTableAssociation:
    Type: AWS::EC2::SubnetRouteTableAssociation
    Properties:
      RouteTableId: !Ref PrivateRouteTable2
      SubnetId: !Ref PrivateSubnet2

  # Security Groups
  ALBSecurityGroup:
    Type: AWS::EC2::SecurityGroup
    Properties:
      GroupName: !Sub '${AWS::StackName}-alb-sg'
      GroupDescription: Security group for Application Load Balancer
      VpcId: !Ref VPC
      SecurityGroupIngress:
        - IpProtocol: tcp
          FromPort: 80
          ToPort: 80
          CidrIp: 0.0.0.0/0
          Description: Allow HTTP traffic from internet
      SecurityGroupEgress:
        - IpProtocol: -1
          CidrIp: 0.0.0.0/0
          Description: Allow all outbound traffic
      Tags:
```

```
      - Key: Name
        Value: !Sub '${AWS::StackName}-alb-sg'

ECSSecurityGroup:
  Type: AWS::EC2::SecurityGroup
  Properties:
    GroupName: !Sub '${AWS::StackName}-ecs-sg'
    GroupDescription: Security group for ECS tasks
    VpcId: !Ref VPC
    SecurityGroupIngress:
      - IpProtocol: tcp
        FromPort: !Ref ContainerPort
        ToPort: !Ref ContainerPort
        SourceSecurityGroupId: !Ref ALBSecurityGroup
        Description: Allow traffic from ALB
    SecurityGroupEgress:
      - IpProtocol: -1
        CidrIp: 0.0.0.0/0
        Description: Allow all outbound traffic
    Tags:
      - Key: Name
        Value: !Sub '${AWS::StackName}-ecs-sg'

# Application Load Balancer
ApplicationLoadBalancer:
  Type: AWS::ElasticLoadBalancingV2::LoadBalancer
  Properties:
    Name: !Sub '${AWS::StackName}-alb'
    Scheme: internet-facing
    Type: application
    Subnets:
      - !Ref PublicSubnet1
      - !Ref PublicSubnet2
    SecurityGroups:
      - !Ref ALBSecurityGroup
    Tags:
      - Key: Name
        Value: !Sub '${AWS::StackName}-alb'

ALBTargetGroup:
  Type: AWS::ElasticLoadBalancingV2::TargetGroup
  Properties:
    Name: !Sub '${AWS::StackName}-tg'
    Port: !Ref ContainerPort
```

```
      Protocol: HTTP
      VpcId: !Ref VPC
      TargetType: ip
      HealthCheckIntervalSeconds: 30
      HealthCheckPath: /
      HealthCheckProtocol: HTTP
      HealthCheckTimeoutSeconds: 5
      HealthyThresholdCount: 2
      UnhealthyThresholdCount: 5
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-tg'

  ALBListener:
    Type: AWS::ElasticLoadBalancingV2::Listener
    Properties:
      DefaultActions:
        - Type: forward
          TargetGroupArn: !Ref ALBTargetGroup
      LoadBalancerArn: !Ref ApplicationLoadBalancer
      Port: 80
      Protocol: HTTP

  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Sub '${AWS::StackName}-cluster'
      CapacityProviders:
        - FARGATE
        - FARGATE_SPOT
      DefaultCapacityProviderStrategy:
        - CapacityProvider: FARGATE
          Weight: 1
        - CapacityProvider: FARGATE_SPOT
          Weight: 4
      ClusterSettings:
        - Name: containerInsights
          Value: enabled
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-cluster'

  # IAM Roles
```

```
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub '${AWS::StackName}-task-execution-role'
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task-execution-role'

  ECSTaskRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: !Sub '${AWS::StackName}-task-role'
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task-role'

  # CloudWatch Log Group
  LogGroup:
    Type: AWS::Logs::LogGroup
    Properties:
      LogGroupName: !Sub '/ecs/${AWS::StackName}'
      RetentionInDays: 7

  # ECS Task Definition
  TaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      Family: !Sub '${AWS::StackName}-task'
```

```
      Cpu: '256'
      Memory: '512'
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - FARGATE
      ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
      TaskRoleArn: !GetAtt ECSTaskRole.Arn
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Image: !Ref ContainerImage
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              Protocol: tcp
          Essential: true
          LogConfiguration:
            LogDriver: awslogs
            Options:
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: ecs
          HealthCheck:
            Command:
              - CMD-SHELL
              - curl -f http://localhost/ || exit 1
            Interval: 30
            Timeout: 5
            Retries: 3
            StartPeriod: 60
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-task'

  # ECS Service
  ECSService:
    Type: AWS::ECS::Service
    DependsOn: ALBListener
    Properties:
      ServiceName: !Sub '${AWS::StackName}-service'
      Cluster: !Ref ECSCluster
      TaskDefinition: !Ref TaskDefinition
      DesiredCount: !Ref DesiredCount
      LaunchType: FARGATE
      PlatformVersion: LATEST
      NetworkConfiguration:
```

```
        AwsvpcConfiguration:
          AssignPublicIp: DISABLED
          SecurityGroups:
            - !Ref ECSSecurityGroup
          Subnets:
            - !Ref PrivateSubnet1
            - !Ref PrivateSubnet2
      LoadBalancers:
        - ContainerName: !Ref ServiceName
          ContainerPort: !Ref ContainerPort
          TargetGroupArn: !Ref ALBTargetGroup
      DeploymentConfiguration:
        MaximumPercent: 200
        MinimumHealthyPercent: 50
        DeploymentCircuitBreaker:
          Enable: true
          Rollback: true
      EnableExecuteCommand: true  # For debugging
      Tags:
        - Key: Name
          Value: !Sub '${AWS::StackName}-service'

  # Auto Scaling Target
  ServiceScalingTarget:
    Type: AWS::ApplicationAutoScaling::ScalableTarget
    Properties:
      MaxCapacity: !Ref MaxCapacity
      MinCapacity: !Ref MinCapacity
      ResourceId: !Sub 'service/${ECSCluster}/${ECSService.Name}'
      RoleARN: !Sub 'arn:aws:iam::${AWS::AccountId}:role/
aws-service-role/ecs.application-autoscaling.amazonaws.com/
AWSServiceRoleForApplicationAutoScaling_ECSService'
      ScalableDimension: ecs:service:DesiredCount
      ServiceNamespace: ecs

  # Auto Scaling Policy - CPU Utilization
  ServiceScalingPolicy:
    Type: AWS::ApplicationAutoScaling::ScalingPolicy
    Properties:
      PolicyName: !Sub '${AWS::StackName}-cpu-scaling-policy'
      PolicyType: TargetTrackingScaling
      ScalingTargetId: !Ref ServiceScalingTarget
      TargetTrackingScalingPolicyConfiguration:
        PredefinedMetricSpecification:
```

```
          PredefinedMetricType: ECSServiceAverageCPUUtilization
        TargetValue: 70.0
        ScaleOutCooldown: 300
        ScaleInCooldown: 300

Outputs:
  VPCId:
    Description: VPC ID
    Value: !Ref VPC
    Export:
      Name: !Sub '${AWS::StackName}-VPC-ID'

  LoadBalancerURL:
    Description: URL of the Application Load Balancer
    Value: !Sub 'http://${ApplicationLoadBalancer.DNSName}'
    Export:
      Name: !Sub '${AWS::StackName}-ALB-URL'

  ECSClusterName:
    Description: Name of the ECS Cluster
    Value: !Ref ECSCluster
    Export:
      Name: !Sub '${AWS::StackName}-ECS-Cluster'

  ECSServiceName:
    Description: Name of the ECS Service
    Value: !GetAtt ECSService.Name
    Export:
      Name: !Sub '${AWS::StackName}-ECS-Service'

  PrivateSubnet1:
    Description: Private Subnet 1 ID
    Value: !Ref PrivateSubnet1
    Export:
      Name: !Sub '${AWS::StackName}-Private-Subnet-1'

  PrivateSubnet2:
    Description: Private Subnet 2 ID
    Value: !Ref PrivateSubnet2
    Export:
      Name: !Sub '${AWS::StackName}-Private-Subnet-2'
```

Déploiement d'un service avec ECS Exec activé

Vous pouvez utiliser le modèle suivant pour déployer un service avec ECS Exec activé. Le service s'exécute dans un cluster avec une clé KMS pour chiffrer les sessions ECS Exec et une configuration de journal pour rediriger les journaux des sessions de commande d'exécution vers un compartiment Amazon S3. Pour de plus amples informations, veuillez consulter Surveillance des conteneurs Amazon ECS avec ECS Exec.

JSON

```
{
    "AWSTemplateFormatVersion": "2010-09-09",
    "Description": "ECS Cluster with Fargate Service and Task Definition and ECS
 Exec enabled.",
    "Parameters": {
        "ClusterName": {
            "Type": "String",
            "Default": "CFNCluster",
            "Description": "Name of the ECS Cluster"
        },
        "S3BucketName": {
            "Type": "String",
            "Default": "amzn-s3-demo-bucket",
            "Description": "S3 bucket for ECS execute command logs"
        },
        "KmsKeyId": {
            "Type": "String",
            "Default": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
            "Description": "KMS Key ID for ECS execute command encryption"
        },
        "ContainerImage": {
            "Type": "String",
            "Default": "public.ecr.aws/docker/library/httpd:2.4",
            "Description": "Container image to use for the task"
        },
        "ContainerCpu": {
            "Type": "Number",
            "Default": 256,
            "AllowedValues": [256, 512, 1024, 2048, 4096],
            "Description": "CPU units for the container (256 = 0.25 vCPU)"
        },
        "ContainerMemory": {
            "Type": "Number",
```

```
            "Default": 512,
            "AllowedValues": [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192],
            "Description": "Memory for the container (in MiB)"
        },
        "DesiredCount": {
            "Type": "Number",
            "Default": 1,
            "Description": "Desired count of tasks in the service"
        },
       "SecurityGroups": {
 "Type": "List<AWS::EC2::SecurityGroup::Id>",
 "Description": "Security Group IDs for the ECS Service"
},
"Subnets": {
 "Type": "List<AWS::EC2::Subnet::Id>",
 "Description": "Subnet IDs for the ECS Service"
},
        "ServiceName": {
            "Type": "String",
            "Default": "cfn-service",
            "Description": "Name of the ECS service"
        },
        "TaskFamily": {
            "Type": "String",
            "Default": "task-definition-cfn",
            "Description": "Family name for the task definition"
        },
        "TaskExecutionRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS task execution",
            "Default": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
        },
        "TaskRoleArn": {
            "Type": "String",
            "Description": "ARN of an existing IAM role for ECS tasks",
            "Default": "arn:aws:iam::111122223333:role/execTaskRole"
        }
    },
    "Resources": {
        "ECSCluster": {
            "Type": "AWS::ECS::Cluster",
            "Properties": {
                "ClusterName": {"Ref": "ClusterName"},
                "Configuration": {
```

```
                    "ExecuteCommandConfiguration": {
                        "Logging": "OVERRIDE",
                        "LogConfiguration": {
                            "S3BucketName": {"Ref": "S3BucketName"}
                        },
                        "KmsKeyId": {"Ref": "KmsKeyId"}
                    }
                },
                "Tags": [
                    {
                        "Key": "Environment",
                        "Value": {"Ref": "AWS::StackName"}
                    }
                ]
            }
        },
        "ECSTaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "ContainerDefinitions": [
                    {
                        "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\""
                        ],
                        "EntryPoint": [
                            "sh",
                            "-c"
                        ],
                        "Essential": true,
                        "Image": {"Ref": "ContainerImage"},
                        "LogConfiguration": {
                            "LogDriver": "awslogs",
                            "Options": {
                                "mode": "non-blocking",
                                "max-buffer-size": "25m",
                                "awslogs-create-group": "true",
                                "awslogs-group": {"Fn::Sub": "/ecs/
${AWS::StackName}"},
                                "awslogs-region": {"Ref": "AWS::Region"},
```

```
                            "awslogs-stream-prefix": "ecs"
                        }
                    },
                    "Name": "sample-fargate-app",
                    "PortMappings": [
                        {
                            "ContainerPort": 80,
                            "HostPort": 80,
                            "Protocol": "tcp"
                        }
                    ]
                }
            ],
            "Cpu": {"Ref": "ContainerCpu"},
            "ExecutionRoleArn": {"Ref": "TaskExecutionRoleArn"},
            "TaskRoleArn": {"Ref": "TaskRoleArn"},
            "Family": {"Ref": "TaskFamily"},
            "Memory": {"Ref": "ContainerMemory"},
            "NetworkMode": "awsvpc",
            "RequiresCompatibilities": [
                "FARGATE"
            ],
            "RuntimePlatform": {
                "OperatingSystemFamily": "LINUX"
            },
            "Tags": [
                {
                    "Key": "Name",
                    "Value": {"Fn::Sub": "${AWS::StackName}-TaskDefinition"}
                }
            ]
        }
    },
    "ECSService": {
        "Type": "AWS::ECS::Service",
        "Properties": {
            "ServiceName": {"Ref": "ServiceName"},
            "Cluster": {"Ref": "ECSCluster"},
            "DesiredCount": {"Ref": "DesiredCount"},
            "EnableExecuteCommand": true,
            "LaunchType": "FARGATE",
            "NetworkConfiguration": {
                "AwsvpcConfiguration": {
                    "AssignPublicIp": "ENABLED",
```

```
                        "SecurityGroups": {"Ref": "SecurityGroups"},
                        "Subnets": {"Ref": "Subnets"}
                    }
                },
                "TaskDefinition": {"Ref": "ECSTaskDefinition"},
                "Tags": [
                    {
                        "Key": "Name",
                        "Value": {"Fn::Sub": "${AWS::StackName}-Service"}
                    }
                ]
            }
        }
    },
    "Outputs": {
        "ClusterName": {
            "Description": "The name of the ECS cluster",
            "Value": {"Ref": "ECSCluster"}
        },
        "ServiceName": {
            "Description": "The name of the ECS service",
            "Value": {"Ref": "ServiceName"}
        },
        "TaskDefinitionArn": {
            "Description": "The ARN of the task definition",
            "Value": {"Ref": "ECSTaskDefinition"}
        },
        "ClusterArn": {
            "Description": "The ARN of the ECS cluster",
            "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
        },
        "StackName": {
            "Description": "The name of this stack",
            "Value": {"Ref": "AWS::StackName"}
        },
        "StackId": {
            "Description": "The unique identifier for this stack",
            "Value": {"Ref": "AWS::StackId"}
        },
        "Region": {
            "Description": "The AWS Region where the stack is deployed",
            "Value": {"Ref": "AWS::Region"}
        },
        "AccountId": {
```

```
            "Description": "The AWS Account ID",
            "Value": {"Ref": "AWS::AccountId"}
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: ECS Cluster with Fargate Service and Task Definition and ECS Exec
 enabled.
Parameters:
  ClusterName:
    Type: String
    Default: CFNCluster
    Description: Name of the ECS Cluster
  S3BucketName:
    Type: String
    Default: amzn-s3-demo-bucket
    Description: S3 bucket for ECS execute command logs
  KmsKeyId:
    Type: String
    Default: a1b2c3d4-5678-90ab-cdef-EXAMPLE11111
    Description: KMS Key ID for ECS execute command encryption
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
  ContainerCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the container (256 = 0.25 vCPU)
  ContainerMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 3072, 4096, 5120, 6144, 7168, 8192]
    Description: Memory for the container (in MiB)
  DesiredCount:
    Type: Number
    Default: 1
    Description: Desired count of tasks in the service
 SecurityGroups:
```

```
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  Subnets:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  ServiceName:
    Type: String
    Default: cfn-service
    Description: Name of the ECS service
  TaskFamily:
    Type: String
    Default: task-definition-cfn
    Description: Family name for the task definition
  TaskExecutionRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS task execution
    Default: 'arn:aws:iam::111122223333:role/ecsTaskExecutionRole'
  TaskRoleArn:
    Type: String
    Description: ARN of an existing IAM role for ECS tasks
    Default: 'arn:aws:iam::111122223333:role/execTaskRole'
Resources:
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ClusterName
      Configuration:
        ExecuteCommandConfiguration:
          Logging: OVERRIDE
          LogConfiguration:
            S3BucketName: !Ref S3BucketName
          KmsKeyId: !Ref KmsKeyId
      Tags:
        - Key: Environment
          Value: !Ref AWS::StackName
  ECSTaskDefinition:
    Type: AWS::ECS::TaskDefinition
    Properties:
      ContainerDefinitions:
        - 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"
        EntryPoint:
          - sh
          - '-c'
        Essential: true
        Image: !Ref ContainerImage
        LogConfiguration:
          LogDriver: awslogs
          Options:
            mode: non-blocking
            max-buffer-size: 25m
            awslogs-create-group: 'true'
            awslogs-group: !Sub /ecs/${AWS::StackName}
            awslogs-region: !Ref AWS::Region
            awslogs-stream-prefix: ecs
        Name: sample-fargate-app
        PortMappings:
          - ContainerPort: 80
            HostPort: 80
            Protocol: tcp
    Cpu: !Ref ContainerCpu
    ExecutionRoleArn: !Ref TaskExecutionRoleArn
    TaskRoleArn: !Ref TaskRoleArn
    Family: !Ref TaskFamily
    Memory: !Ref ContainerMemory
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX
    Tags:
      - Key: Name
        Value: !Sub ${AWS::StackName}-TaskDefinition
ECSService:
  Type: AWS::ECS::Service
  Properties:
    ServiceName: !Ref ServiceName
    Cluster: !Ref ECSCluster
    DesiredCount: !Ref DesiredCount
    EnableExecuteCommand: true
    LaunchType: FARGATE
```

```
      NetworkConfiguration:
        AwsvpcConfiguration:
          AssignPublicIp: ENABLED
          SecurityGroups: !Ref SecurityGroups
          Subnets: !Ref Subnets
      TaskDefinition: !Ref ECSTaskDefinition
      Tags:
        - Key: Name
          Value: !Sub ${AWS::StackName}-Service
Outputs:
  ClusterName:
    Description: The name of the ECS cluster
    Value: !Ref ECSCluster
  ServiceName:
    Description: The name of the ECS service
    Value: !Ref ServiceName
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref ECSTaskDefinition
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  StackName:
    Description: The name of this stack
    Value: !Ref AWS::StackName
  StackId:
    Description: The unique identifier for this stack
    Value: !Ref AWS::StackId
  Region:
    Description: The AWS Region where the stack is deployed
    Value: !Ref AWS::Region
  AccountId:
    Description: The AWS Account ID
    Value: !Ref AWS::AccountId
```

Déploiement d’un service qui utilise Amazon VPC Lattice

Vous pouvez utiliser le modèle suivant pour commencer à créer un service Amazon ECS avec VPC Lattice. Vous devrez peut-être effectuer les étapes supplémentaires suivantes pour configurer VPC Lattice :

- Mettez à jour les règles entrantes de votre groupe de sécurité pour VPC Lattice afin d'autoriser le préfixe `vpc-lattice` de règle entrante et d'autoriser le trafic sur le port 80.
- Associez le VPC du service à un réseau de services VPC Lattice.
- Configurez une zone hébergée privée ou publique avec Amazon Route 53.
- Configurez les écouteurs et les règles d'écoute dans un service VPC Lattice.
- Vérifiez les configurations de surveillance de l'état du groupe cible.

Pour plus d'informations sur l'utilisation de VPC Lattice avec Amazon ECS, consultez la section Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS.

JSON

```
{
 "AWSTemplateFormatVersion": "2010-09-09",
 "Description": "The template used to create an ECS Service with VPC Lattice.",
 "Parameters": {
  "ECSClusterName": {
   "Type": "String",
   "Default": "vpc-lattice-cluster"
  },
  "ECSServiceName": {
   "Type": "String",
   "Default": "vpc-lattice-service"
  },
  "SecurityGroupIDs": {
   "Type": "List<AWS::EC2::SecurityGroup::Id>",
   "Description": "Security Group IDs for the ECS Service"
  },
  "SubnetIDs": {
   "Type": "List<AWS::EC2::Subnet::Id>",
   "Description": "Subnet IDs for the ECS Service"
  },
  "VpcID": {
   "Type": "AWS::EC2::VPC::Id",
   "Description": "VPC ID for the resources"
  },
  "ContainerImage": {
   "Type": "String",
   "Default": "public.ecr.aws/docker/library/httpd:2.4",
   "Description": "Container image to use for the task"
  },
```

```
  "TaskCpu": {
   "Type": "Number",
   "Default": 256,
   "AllowedValues": [256, 512, 1024, 2048, 4096],
   "Description": "CPU units for the task"
  },
  "TaskMemory": {
   "Type": "Number",
   "Default": 512,
   "AllowedValues": [512, 1024, 2048, 4096, 8192, 16384],
   "Description": "Memory (in MiB) for the task"
  },
  "LogGroupName": {
   "Type": "String",
   "Default": "/ecs/vpc-lattice-task",
   "Description": "CloudWatch Log Group name"
  },
  "EnableContainerInsights": {
   "Type": "String",
   "Default": "enabled",
   "AllowedValues": ["enabled", "disabled"],
   "Description": "Enable or disable CloudWatch Container Insights for the cluster"
  }
 },
 "Resources": {
  "ECSCluster": {
   "Type": "AWS::ECS::Cluster",
   "Properties": {
    "ClusterName": {"Ref": "ECSClusterName"},
    "ClusterSettings": [
     {
      "Name": "containerInsights",
      "Value": {"Ref": "EnableContainerInsights"}
     }
    ],
    "Tags": [
     {
      "Key": "Name",
      "Value": {"Ref": "ECSClusterName"}
     }
    ]
   }
  },
  "ECSTaskExecutionRole": {
```

```
  "Type": "AWS::IAM::Role",
  "Properties": {
   "AssumeRolePolicyDocument": {
    "Version": "2012-10-17",
    "Statement": [
     {
      "Effect": "Allow",
      "Principal": {
       "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
     }
    ]
   },
   "ManagedPolicyArns": [
    "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
   ]
  }
 },
 "TaskLogGroup": {
  "Type": "AWS::Logs::LogGroup",
  "DeletionPolicy": "Retain",
  "UpdateReplacePolicy": "Retain",
  "Properties": {
   "LogGroupName": {"Ref": "LogGroupName"},
   "RetentionInDays": 30
  }
 },
 "VpcLatticeTaskDefinition": {
  "Type": "AWS::ECS::TaskDefinition",
  "Properties": {
   "ContainerDefinitions": [
    {
     "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\""
     ],
     "EntryPoint": [
      "sh",
      "-c"
```

```
      ],
      "Essential": true,
      "Image": {"Ref": "ContainerImage"},
      "LogConfiguration": {
       "LogDriver": "awslogs",
       "Options": {
        "mode": "non-blocking",
        "max-buffer-size": "25m",
        "awslogs-create-group": "true",
        "awslogs-group": {"Ref": "LogGroupName"},
        "awslogs-region": {"Ref": "AWS::Region"},
        "awslogs-stream-prefix": "ecs"
       }
      },
      "Name": "vpc-lattice-container",
      "PortMappings": [
       {
        "ContainerPort": 80,
        "HostPort": 80,
        "Protocol": "tcp",
        "Name": "vpc-lattice-port"
       }
      ]
     }
    ],
    "Cpu": {"Ref": "TaskCpu"},
    "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
    "Family": "vpc-lattice-task-definition",
    "Memory": {"Ref": "TaskMemory"},
    "NetworkMode": "awsvpc",
    "RequiresCompatibilities": [
     "FARGATE"
    ],
    "RuntimePlatform": {
     "OperatingSystemFamily": "LINUX"
    }
   }
  },
  "ECSService": {
   "Type": "AWS::ECS::Service",
   "Properties": {
    "Cluster": {"Ref": "ECSCluster"},
    "TaskDefinition": {"Ref": "VpcLatticeTaskDefinition"},
    "LaunchType": "FARGATE",
```

```
    "ServiceName": {"Ref": "ECSServiceName"},
    "SchedulingStrategy": "REPLICA",
    "DesiredCount": 2,
    "AvailabilityZoneRebalancing": "ENABLED",
    "NetworkConfiguration": {
     "AwsvpcConfiguration": {
      "AssignPublicIp": "ENABLED",
      "SecurityGroups": {
       "Ref": "SecurityGroupIDs"
      },
      "Subnets": {
       "Ref": "SubnetIDs"
      }
     }
    },
    "PlatformVersion": "LATEST",
    "VpcLatticeConfigurations": [
     {
      "RoleArn": "arn:aws:iam::111122223333:role/ecsInfrastructureRole",
      "PortName": "vpc-lattice-port",
      "TargetGroupArn": {
       "Ref": "TargetGroup1"
      }
     }
    ],
    "DeploymentConfiguration": {
     "DeploymentCircuitBreaker": {
      "Enable": true,
      "Rollback": true
     },
     "MaximumPercent": 200,
     "MinimumHealthyPercent": 100
    },
    "DeploymentController": {
     "Type": "ECS"
    },
    "ServiceConnectConfiguration": {
     "Enabled": false
    },
    "Tags": [],
    "EnableECSManagedTags": true
   }
  },
  "TargetGroup1": {
```

```
   "Type": "AWS::VpcLattice::TargetGroup",
   "Properties": {
    "Type": "IP",
    "Name": "first-target-group",
    "Config": {
     "Port": 80,
     "Protocol": "HTTP",
     "VpcIdentifier": {"Ref": "VpcID"},
     "HealthCheck": {
      "Enabled": true,
      "Path": "/"
     }
    },
    "Tags": [
     {
      "Key": "ecs-application-networking/ServiceName",
      "Value": {"Ref": "ECSServiceName"}
     },
     {
      "Key": "ecs-application-networking/ClusterName",
      "Value": {"Ref": "ECSClusterName"}
     },
     {
      "Key": "ecs-application-networking/TaskDefinition",
      "Value": {"Ref": "VpcLatticeTaskDefinition"}
     },
     {
      "Key": "ecs-application-networking/VpcId",
      "Value": {"Ref": "VpcID"}
     }
    ]
   }
  }
 },
 "Outputs": {
  "ClusterName": {
   "Description": "The cluster used to create the service.",
   "Value": {
    "Ref": "ECSCluster"
   }
  },
  "ClusterArn": {
   "Description": "The ARN of the ECS cluster",
   "Value": {
```

```
    "Fn::GetAtt": ["ECSCluster", "Arn"]
   }
  },
  "ECSService": {
   "Description": "The created service.",
   "Value": {
    "Ref": "ECSService"
   }
  },
  "TaskDefinitionArn": {
   "Description": "The ARN of the task definition",
   "Value": {
    "Ref": "VpcLatticeTaskDefinition"
   }
  }
 }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: The template used to create an ECS Service with VPC Lattice.

Parameters:
  ECSClusterName:
    Type: String
    Default: vpc-lattice-cluster
  ECSServiceName:
    Type: String
    Default: vpc-lattice-service
  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service
  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service
  VpcID:
    Type: AWS::EC2::VPC::Id
    Description: VPC ID for the resources
  ContainerImage:
    Type: String
    Default: public.ecr.aws/docker/library/httpd:2.4
    Description: Container image to use for the task
```

```
  TaskCpu:
    Type: Number
    Default: 256
    AllowedValues: [256, 512, 1024, 2048, 4096]
    Description: CPU units for the task
  TaskMemory:
    Type: Number
    Default: 512
    AllowedValues: [512, 1024, 2048, 4096, 8192, 16384]
    Description: Memory (in MiB) for the task
  LogGroupName:
    Type: String
    Default: /ecs/vpc-lattice-task
    Description: CloudWatch Log Group name
  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
  ECSCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: !Ref ECSClusterName
      ClusterSettings:
        - Name: containerInsights
          Value: !Ref EnableContainerInsights
      Tags:
        - Key: Name
          Value: !Ref ECSClusterName

  # IAM Roles
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Version: '2012-10-17       '
        Statement:
          - Effect: Allow
            Principal:
              Service: ecs-tasks.amazonaws.com
            Action: sts:AssumeRole
```

```
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

# CloudWatch Logs
TaskLogGroup:
  Type: AWS::Logs::LogGroup
  DeletionPolicy: Retain
  UpdateReplacePolicy: Retain
  Properties:
    LogGroupName: !Ref LogGroupName
    RetentionInDays: 30

# Task Definition
VpcLatticeTaskDefinition:
  Type: AWS::ECS::TaskDefinition
  Properties:
    ContainerDefinitions:
      - 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"
        EntryPoint:
          - sh
          - '-c'
        Essential: true
        Image: !Ref ContainerImage
        LogConfiguration:
          LogDriver: awslogs
          Options:
            mode: non-blocking
            max-buffer-size: 25m
            awslogs-create-group: 'true'
            awslogs-group: !Ref LogGroupName
            awslogs-region: !Ref 'AWS::Region'
            awslogs-stream-prefix: ecs
        Name: vpc-lattice-container
        PortMappings:
          - ContainerPort: 80
            HostPort: 80
```

```
            Protocol: tcp
            Name: vpc-lattice-port
    Cpu: !Ref TaskCpu
    ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
    Family: vpc-lattice-task-definition
    Memory: !Ref TaskMemory
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - FARGATE
    RuntimePlatform:
      OperatingSystemFamily: LINUX

ECSService:
  Type: AWS::ECS::Service
  Properties:
    Cluster: !Ref ECSCluster
    TaskDefinition: !Ref VpcLatticeTaskDefinition
    LaunchType: FARGATE
    ServiceName: !Ref ECSServiceName
    SchedulingStrategy: REPLICA
    DesiredCount: 2
    AvailabilityZoneRebalancing: ENABLED
    NetworkConfiguration:
      AwsvpcConfiguration:
        AssignPublicIp: ENABLED
        SecurityGroups: !Ref SecurityGroupIDs
        Subnets: !Ref SubnetIDs
    PlatformVersion: LATEST
    VpcLatticeConfigurations:
      - RoleArn: arn:aws:iam::111122223333:role/ecsInfrastructureRole
        PortName: vpc-lattice-port
        TargetGroupArn: !Ref TargetGroup1
    DeploymentConfiguration:
      DeploymentCircuitBreaker:
        Enable: true
        Rollback: true
      MaximumPercent: 200
      MinimumHealthyPercent: 100
    DeploymentController:
      Type: ECS
    ServiceConnectConfiguration:
      Enabled: false
    Tags: []
    EnableECSManagedTags: true
```

```
  TargetGroup1:
    Type: AWS::VpcLattice::TargetGroup
    Properties:
      Type: IP
      Name: first-target-group
      Config:
        Port: 80
        Protocol: HTTP
        VpcIdentifier: !Ref VpcID
        HealthCheck:
          Enabled: true
          Path: /
      Tags:
        - Key: ecs-application-networking/ServiceName
          Value: !Ref ECSServiceName
        - Key: ecs-application-networking/ClusterName
          Value: !Ref ECSClusterName
        - Key: ecs-application-networking/TaskDefinition
          Value: !Ref VpcLatticeTaskDefinition
        - Key: ecs-application-networking/VpcId
          Value: !Ref VpcID

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref VpcLatticeTaskDefinition
```

Déploiement d’un service avec une configuration de volume

Le modèle suivant inclut une configuration de volume dans la définition du service. Amazon ECS prend en charge la configuration des volumes de données suivants en utilisant une configuration de

volume au lancement : volumes Amazon EBS. Pour plus d'informations sur les volumes Amazon EBS, reportez-vous à la section Utilisation des volumes Amazon EBS avec Amazon ECS.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "The template used to create an ECS Service that includes a
 volume configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.",
  "Parameters": {
    "ECSClusterName": {
      "Type": "String",
      "Default": "volume-config-cluster",
      "Description": "Name of the ECS cluster"
    },
    "SecurityGroupIDs": {
      "Type": "List<AWS::EC2::SecurityGroup::Id>",
      "Description": "Security Group IDs for the ECS Service"
    },
    "SubnetIDs": {
      "Type": "List<AWS::EC2::Subnet::Id>",
      "Description": "Subnet IDs for the ECS Service"
    },
    "InfrastructureRoleArn": {
      "Type": "String",
      "Description": "ARN of the IAM role that ECS will use to manage EBS volumes"
    },
    "ContainerImage": {
      "Type": "String",
      "Default": "public.ecr.aws/nginx/nginx:latest",
      "Description": "Container image to use for the task"
    },
    "TaskCpu": {
      "Type": "String",
      "Default": "2048",
      "Description": "CPU units for the task"
    },
    "TaskMemory": {
      "Type": "String",
      "Default": "4096",
      "Description": "Memory (in MiB) for the task"
    },
```

```
    "VolumeSize": {
      "Type": "String",
      "Default": "10",
      "Description": "Size of the EBS volume in GiB"
    },
    "VolumeType": {
      "Type": "String",
      "Default": "gp3",
      "AllowedValues": ["gp2", "gp3", "io1", "io2", "st1", "sc1", "standard"],
      "Description": "EBS volume type"
    },
    "VolumeIops": {
      "Type": "String",
      "Default": "3000",
      "Description": "IOPS for the EBS volume (required for io1, io2, and gp3)"
    },
    "VolumeThroughput": {
      "Type": "String",
      "Default": "125",
      "Description": "Throughput for the EBS volume (only for gp3)"
    },
    "FilesystemType": {
      "Type": "String",
      "Default": "xfs",
      "AllowedValues": ["xfs", "ext4"],
      "Description": "Filesystem type for the EBS volume"
    },
    "EnableContainerInsights": {
      "Type": "String",
      "Default": "enhanced",
      "AllowedValues": ["enabled", "disabled", "enhanced"],
      "Description": "Enable or disable CloudWatch Container Insights for the
cluster"
    }
  },
  "Resources": {
    "ECSCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {
        "ClusterName": {"Ref": "ECSClusterName"},
        "ClusterSettings": [
          {
            "Name": "containerInsights",
            "Value": {"Ref": "EnableContainerInsights"}
```

```
          }
        ],
        "Tags": [
          {
            "Key": "Name",
            "Value": {"Ref": "ECSClusterName"}
          }
        ]
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    },
    "EBSTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "Family": "ebs-task-attach-task-def",
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "NetworkMode": "awsvpc",
        "RequiresCompatibilities": [
          "EC2",
          "FARGATE"
        ],
        "Cpu": {"Ref": "TaskCpu"},
        "Memory": {"Ref": "TaskMemory"},
        "ContainerDefinitions": [
          {
```

```
            "Name": "nginx",
            "Image": {"Ref": "ContainerImage"},
            "Essential": true,
            "PortMappings": [
              {
                "Name": "nginx-80-tcp",
                "ContainerPort": 80,
                "HostPort": 80,
                "Protocol": "tcp",
                "AppProtocol": "http"
              }
            ],
            "MountPoints": [
              {
                "SourceVolume": "ebs-vol",
                "ContainerPath": "/foo-container-path",
                "ReadOnly": false
              }
            ]
          }
        ],
        "Volumes": [
          {
            "Name": "ebs-vol",
            "ConfiguredAtLaunch": true
          }
        ]
      }
    },
    "ECSService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSCluster"},
        "TaskDefinition": {"Ref": "EBSTaskDefinition"},
        "LaunchType": "FARGATE",
        "ServiceName": "ebs",
        "SchedulingStrategy": "REPLICA",
        "DesiredCount": 1,
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": {"Ref": "SecurityGroupIDs"},
            "Subnets": {"Ref": "SubnetIDs"}
          }
```

```
        },
        "PlatformVersion": "LATEST",
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          }
        },
        "DeploymentController": {
          "Type": "ECS"
        },
        "Tags": [],
        "EnableECSManagedTags": true,
        "VolumeConfigurations": [
          {
            "Name": "ebs-vol",
            "ManagedEBSVolume": {
              "RoleArn": {"Ref": "InfrastructureRoleArn"},
              "VolumeType": {"Ref": "VolumeType"},
              "Iops": {"Ref": "VolumeIops"},
              "Throughput": {"Ref": "VolumeThroughput"},
              "SizeInGiB": {"Ref": "VolumeSize"},
              "FilesystemType": {"Ref": "FilesystemType"},
              "TagSpecifications": [
                {
                  "ResourceType": "volume",
                  "PropagateTags": "TASK_DEFINITION"
                }
              ]
            }
          }
        ]
      }
    }
  },
  "Outputs": {
    "ClusterName": {
      "Description": "The cluster used to create the service.",
      "Value": {"Ref": "ECSCluster"}
    },
    "ClusterArn": {
      "Description": "The ARN of the ECS cluster",
```

```
      "Value": {"Fn::GetAtt": ["ECSCluster", "Arn"]}
    },
    "ECSService": {
      "Description": "The created service.",
      "Value": {"Ref": "ECSService"}
    },
    "TaskDefinitionArn": {
      "Description": "The ARN of the task definition",
      "Value": {"Ref": "EBSTaskDefinition"}
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: 2010-09-09
Description: The template used to create an ECS Service that includes a volume
 configuration. The configuration is used to create Amazon EBS volumes for
 attachment to the tasks. One volume is attached per task.
Parameters:
  ECSClusterName:
    Type: String
    Default: volume-config-cluster
    Description: Name of the ECS cluster

  SecurityGroupIDs:
    Type: List<AWS::EC2::SecurityGroup::Id>
    Description: Security Group IDs for the ECS Service

  SubnetIDs:
    Type: List<AWS::EC2::Subnet::Id>
    Description: Subnet IDs for the ECS Service

  InfrastructureRoleArn:
    Type: String
    Description: ARN of the IAM role that ECS will use to manage EBS volumes

  ContainerImage:
    Type: String
    Default: public.ecr.aws/nginx/nginx:latest
    Description: Container image to use for the task

  TaskCpu:
```

```
    Type: String
    Default: "2048"
    Description: CPU units for the task

  TaskMemory:
    Type: String
    Default: "4096"
    Description: Memory (in MiB) for the task

  VolumeSize:
    Type: String
    Default: "10"
    Description: Size of the EBS volume in GiB

  VolumeType:
    Type: String
    Default: gp3
    AllowedValues: [gp2, gp3, io1, io2, st1, sc1, standard]
    Description: EBS volume type

  VolumeIops:
    Type: String
    Default: "3000"
    Description: IOPS for the EBS volume (required for io1, io2, and gp3)

  VolumeThroughput:
    Type: String
    Default: "125"
    Description: Throughput for the EBS volume (only for gp3)

  FilesystemType:
    Type: String
    Default: xfs
    AllowedValues: [xfs, ext4]
    Description: Filesystem type for the EBS volume

  EnableContainerInsights:
    Type: String
    Default: 'enhanced'
    AllowedValues: ['enabled', 'disabled', 'enhanced']
    Description: Enable or disable CloudWatch Container Insights for the cluster

Resources:
  # ECS Cluster
```

```
ECSCluster:
  Type: AWS::ECS::Cluster
  Properties:
    ClusterName: !Ref ECSClusterName
    ClusterSettings:
      - Name: containerInsights
        Value: !Ref EnableContainerInsights
    Tags:
      - Key: Name
        Value: !Ref ECSClusterName

# IAM Role for Task Execution
ECSTaskExecutionRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: '2012-10-17       '
      Statement:
        - Effect: Allow
          Principal:
            Service: ecs-tasks.amazonaws.com
          Action: sts:AssumeRole
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy

# Task Definition
EBSTaskDefinition:
  Type: AWS::ECS::TaskDefinition
  Properties:
    Family: ebs-task-attach-task-def
    ExecutionRoleArn: !GetAtt ECSTaskExecutionRole.Arn
    NetworkMode: awsvpc
    RequiresCompatibilities:
      - EC2
      - FARGATE
    Cpu: !Ref TaskCpu
    Memory: !Ref TaskMemory
    ContainerDefinitions:
      - Name: nginx
        Image: !Ref ContainerImage
        Essential: true
        PortMappings:
          - Name: nginx-80-tcp
            ContainerPort: 80
```

```
            HostPort: 80
            Protocol: tcp
            AppProtocol: http
        MountPoints:
          - SourceVolume: ebs-vol
            ContainerPath: /foo-container-path
            ReadOnly: false
    Volumes:
      - Name: ebs-vol
        ConfiguredAtLaunch: true

ECSService:
  Type: AWS::ECS::Service
  Properties:
    Cluster: !Ref ECSCluster
    TaskDefinition: !Ref EBSTaskDefinition
    LaunchType: FARGATE
    ServiceName: ebs
    SchedulingStrategy: REPLICA
    DesiredCount: 1
    NetworkConfiguration:
      AwsvpcConfiguration:
        AssignPublicIp: ENABLED
        SecurityGroups: !Ref SecurityGroupIDs
        Subnets: !Ref SubnetIDs
    PlatformVersion: LATEST
    DeploymentConfiguration:
      MaximumPercent: 200
      MinimumHealthyPercent: 100
      DeploymentCircuitBreaker:
        Enable: true
        Rollback: true
    DeploymentController:
      Type: ECS
    Tags: []
    EnableECSManagedTags: true
    VolumeConfigurations:
      - Name: ebs-vol
        ManagedEBSVolume:
          RoleArn: !Ref InfrastructureRoleArn
          VolumeType: !Ref VolumeType
          Iops: !Ref VolumeIops
          Throughput: !Ref VolumeThroughput
          SizeInGiB: !Ref VolumeSize
```

```
            FilesystemType: !Ref FilesystemType
            TagSpecifications:
              - ResourceType: volume
                PropagateTags: TASK_DEFINITION

Outputs:
  ClusterName:
    Description: The cluster used to create the service.
    Value: !Ref ECSCluster
  ClusterArn:
    Description: The ARN of the ECS cluster
    Value: !GetAtt ECSCluster.Arn
  ECSService:
    Description: The created service.
    Value: !Ref ECSService
  TaskDefinitionArn:
    Description: The ARN of the task definition
    Value: !Ref EBSTaskDefinition
```

Déploiement d’un service avec fournisseurs de capacité

Le modèle suivant définit un service qui utilise le fournisseur de capacité pour demander la capacité AL2023 à exécuter. Les conteneurs seront lancés sur les instances AL2023 dès leur mise en ligne.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "An example service that deploys in AWS VPC networking mode on
 EC2 capacity. Service uses a capacity provider to request EC2 instances to run
 on. Service runs with networking in private subnets, but still accessible to the
 internet via a load balancer hosted in public subnets.",
  "Parameters": {
      "VpcId": {
          "Type": "String",
          "Description": "The VPC that the service is running inside of"
      },
      "PublicSubnetIds": {
          "Type": "List<AWS::EC2::Subnet::Id>",
          "Description": "List of public subnet ID's to put the load balancer in"
      },
      "PrivateSubnetIds": {
```

```
          "Type": "List<AWS::EC2::Subnet::Id>",
          "Description": "List of private subnet ID's that the AWS VPC tasks are in"
      },
      "ClusterName": {
          "Type": "String",
          "Description": "The name of the ECS cluster into which to launch
capacity."
      },
      "ECSTaskExecutionRole": {
          "Type": "String",
          "Description": "The role used to start up an ECS task"
      },
      "CapacityProvider": {
          "Type": "String",
          "Description": "The cluster capacity provider that the service should use
to request capacity when it wants to start up a task"
      },
      "ServiceName": {
          "Type": "String",
          "Default": "web",
          "Description": "A name for the service"
      },
      "ImageUrl": {
          "Type": "String",
          "Default": "public.ecr.aws/docker/library/nginx:latest",
          "Description": "The url of a docker image that contains the application
process that will handle the traffic for this service"
      },
      "ContainerCpu": {
          "Type": "Number",
          "Default": 256,
          "Description": "How much CPU to give the container. 1024 is 1 CPU"
      },
      "ContainerMemory": {
          "Type": "Number",
          "Default": 512,
          "Description": "How much memory in megabytes to give the container"
      },
      "ContainerPort": {
          "Type": "Number",
          "Default": 80,
          "Description": "What port that the application expects traffic on"
      },
      "DesiredCount": {
```

```
            "Type": "Number",
            "Default": 2,
            "Description": "How many copies of the service task to run"
        }
    },
    "Resources": {
        "TaskDefinition": {
            "Type": "AWS::ECS::TaskDefinition",
            "Properties": {
                "Family": {
                    "Ref": "ServiceName"
                },
                "Cpu": {
                    "Ref": "ContainerCpu"
                },
                "Memory": {
                    "Ref": "ContainerMemory"
                },
                "NetworkMode": "awsvpc",
                "RequiresCompatibilities": [
                    "EC2"
                ],
                "ExecutionRoleArn": {
                    "Ref": "ECSTaskExecutionRole"
                },
                "ContainerDefinitions": [
                    {
                        "Name": {
                            "Ref": "ServiceName"
                        },
                        "Cpu": {
                            "Ref": "ContainerCpu"
                        },
                        "Memory": {
                            "Ref": "ContainerMemory"
                        },
                        "Image": {
                            "Ref": "ImageUrl"
                        },
                        "PortMappings": [
                            {
                                "ContainerPort": {
                                    "Ref": "ContainerPort"
                                },
```

```
                            "HostPort": {
                                "Ref": "ContainerPort"
                            }
                        }
                    ],
                    "LogConfiguration": {
                        "LogDriver": "awslogs",
                        "Options": {
                            "mode": "non-blocking",
                            "max-buffer-size": "25m",
                            "awslogs-group": {
                                "Ref": "LogGroup"
                            },
                            "awslogs-region": {
                                "Ref": "AWS::Region"
                            },
                            "awslogs-stream-prefix": {
                                "Ref": "ServiceName"
                            }
                        }
                    }
                }
            ]
        }
    },
    "Service": {
        "Type": "AWS::ECS::Service",
        "DependsOn": "PublicLoadBalancerListener",
        "Properties": {
            "ServiceName": {
                "Ref": "ServiceName"
            },
            "Cluster": {
                "Ref": "ClusterName"
            },
            "PlacementStrategies": [
                {
                    "Field": "attribute:ecs.availability-zone",
                    "Type": "spread"
                },
                {
                    "Field": "cpu",
                    "Type": "binpack"
                }
```

```
            ],
            "CapacityProviderStrategy": [
                {
                    "Base": 0,
                    "CapacityProvider": {
                        "Ref": "CapacityProvider"
                    },
                    "Weight": 1
                }
            ],
            "NetworkConfiguration": {
                "AwsvpcConfiguration": {
                    "SecurityGroups": [
                        {
                            "Ref": "ServiceSecurityGroup"
                        }
                    ],
                    "Subnets": {
                        "Ref": "PrivateSubnetIds"
                    }
                }
            },
            "DeploymentConfiguration": {
                "MaximumPercent": 200,
                "MinimumHealthyPercent": 75
            },
            "DesiredCount": {
                "Ref": "DesiredCount"
            },
            "TaskDefinition": {
                "Ref": "TaskDefinition"
            },
            "LoadBalancers": [
                {
                    "ContainerName": {
                        "Ref": "ServiceName"
                    },
                    "ContainerPort": {
                        "Ref": "ContainerPort"
                    },
                    "TargetGroupArn": {
                        "Ref": "ServiceTargetGroup"
                    }
                }
```

```
            ]
        }
    },
    "ServiceSecurityGroup": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Security group for service",
            "VpcId": {
                "Ref": "VpcId"
            }
        }
    },
    "ServiceTargetGroup": {
        "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
        "Properties": {
            "HealthCheckIntervalSeconds": 6,
            "HealthCheckPath": "/",
            "HealthCheckProtocol": "HTTP",
            "HealthCheckTimeoutSeconds": 5,
            "HealthyThresholdCount": 2,
            "TargetType": "ip",
            "Port": {
                "Ref": "ContainerPort"
            },
            "Protocol": "HTTP",
            "UnhealthyThresholdCount": 10,
            "VpcId": {
                "Ref": "VpcId"
            },
            "TargetGroupAttributes": [
                {
                    "Key": "deregistration_delay.timeout_seconds",
                    "Value": 0
                }
            ]
        }
    },
    "PublicLoadBalancerSG": {
        "Type": "AWS::EC2::SecurityGroup",
        "Properties": {
            "GroupDescription": "Access to the public facing load balancer",
            "VpcId": {
                "Ref": "VpcId"
            },
```

```
            "SecurityGroupIngress": [
                {
                    "CidrIp": "0.0.0.0/0",
                    "IpProtocol": -1
                }
            ]
        }
    },
    "PublicLoadBalancer": {
        "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
        "Properties": {
            "Scheme": "internet-facing",
            "LoadBalancerAttributes": [
                {
                    "Key": "idle_timeout.timeout_seconds",
                    "Value": "30"
                }
            ],
            "Subnets": {
                "Ref": "PublicSubnetIds"
            },
            "SecurityGroups": [
                {
                    "Ref": "PublicLoadBalancerSG"
                }
            ]
        }
    },
    "PublicLoadBalancerListener": {
        "Type": "AWS::ElasticLoadBalancingV2::Listener",
        "Properties": {
            "DefaultActions": [
                {
                    "Type": "forward",
                    "ForwardConfig": {
                        "TargetGroups": [
                            {
                                "TargetGroupArn": {
                                    "Ref": "ServiceTargetGroup"
                                },
                                "Weight": 100
                            }
                        ]
                    }
```

```
                    }
                ],
                "LoadBalancerArn": {
                    "Ref": "PublicLoadBalancer"
                },
                "Port": 80,
                "Protocol": "HTTP"
            }
        },
        "ServiceIngressfromLoadBalancer": {
            "Type": "AWS::EC2::SecurityGroupIngress",
            "Properties": {
                "Description": "Ingress from the public ALB",
                "GroupId": {
                    "Ref": "ServiceSecurityGroup"
                },
                "IpProtocol": -1,
                "SourceSecurityGroupId": {
                    "Ref": "PublicLoadBalancerSG"
                }
            }
        },
        "LogGroup": {
            "Type": "AWS::Logs::LogGroup"
        }
    }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: >-
  An example service that deploys in AWS VPC networking mode on EC2 capacity.
  Service uses a capacity provider to request EC2 instances to run on. Service
  runs with networking in private subnets, but still accessible to the internet
  via a load balancer hosted in public subnets.
Parameters:
  VpcId:
    Type: String
    Description: The VPC that the service is running inside of
  PublicSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of public subnet ID's to put the load balancer in
```

```
  PrivateSubnetIds:
    Type: 'List<AWS::EC2::Subnet::Id>'
    Description: List of private subnet ID's that the AWS VPC tasks are in
  ClusterName:
    Type: String
    Description: The name of the ECS cluster into which to launch capacity.
  ECSTaskExecutionRole:
    Type: String
    Description: The role used to start up an ECS task
  CapacityProvider:
    Type: String
    Description: >-
      The cluster capacity provider that the service should use to request
      capacity when it wants to start up a task
  ServiceName:
    Type: String
    Default: web
    Description: A name for the service
  ImageUrl:
    Type: String
    Default: 'public.ecr.aws/docker/library/nginx:latest'
    Description: >-
      The url of a docker image that contains the application process that will
      handle the traffic for this service
  ContainerCpu:
    Type: Number
    Default: 256
    Description: How much CPU to give the container. 1024 is 1 CPU
  ContainerMemory:
    Type: Number
    Default: 512
    Description: How much memory in megabytes to give the container
  ContainerPort:
    Type: Number
    Default: 80
    Description: What port that the application expects traffic on
  DesiredCount:
    Type: Number
    Default: 2
    Description: How many copies of the service task to run
Resources:
  TaskDefinition:
    Type: 'AWS::ECS::TaskDefinition'
    Properties:
```

```
      Family: !Ref ServiceName
      Cpu: !Ref ContainerCpu
      Memory: !Ref ContainerMemory
      NetworkMode: awsvpc
      RequiresCompatibilities:
        - EC2
      ExecutionRoleArn: !Ref ECSTaskExecutionRole
      ContainerDefinitions:
        - Name: !Ref ServiceName
          Cpu: !Ref ContainerCpu
          Memory: !Ref ContainerMemory
          Image: !Ref ImageUrl
          PortMappings:
            - ContainerPort: !Ref ContainerPort
              HostPort: !Ref ContainerPort
          LogConfiguration:
            LogDriver: awslogs
            Options:
              mode: non-blocking
              max-buffer-size: 25m
              awslogs-group: !Ref LogGroup
              awslogs-region: !Ref AWS::Region
              awslogs-stream-prefix: !Ref ServiceName
  Service:
    Type: AWS::ECS::Service
    DependsOn: PublicLoadBalancerListener
    Properties:
      ServiceName: !Ref ServiceName
      Cluster: !Ref ClusterName
      PlacementStrategies:
        - Field: 'attribute:ecs.availability-zone'
          Type: spread
        - Field: cpu
          Type: binpack
      CapacityProviderStrategy:
        - Base: 0
          CapacityProvider: !Ref CapacityProvider
          Weight: 1
      NetworkConfiguration:
        AwsvpcConfiguration:
          SecurityGroups:
            - !Ref ServiceSecurityGroup
          Subnets: !Ref PrivateSubnetIds
      DeploymentConfiguration:
```

```
      MaximumPercent: 200
      MinimumHealthyPercent: 75
    DesiredCount: !Ref DesiredCount
    TaskDefinition: !Ref TaskDefinition
    LoadBalancers:
      - ContainerName: !Ref ServiceName
        ContainerPort: !Ref ContainerPort
        TargetGroupArn: !Ref ServiceTargetGroup
ServiceSecurityGroup:
  Type: 'AWS::EC2::SecurityGroup'
  Properties:
    GroupDescription: Security group for service
    VpcId: !Ref VpcId
ServiceTargetGroup:
  Type: 'AWS::ElasticLoadBalancingV2::TargetGroup'
  Properties:
    HealthCheckIntervalSeconds: 6
    HealthCheckPath: /
    HealthCheckProtocol: HTTP
    HealthCheckTimeoutSeconds: 5
    HealthyThresholdCount: 2
    TargetType: ip
    Port: !Ref ContainerPort
    Protocol: HTTP
    UnhealthyThresholdCount: 10
    VpcId: !Ref VpcId
    TargetGroupAttributes:
      - Key: deregistration_delay.timeout_seconds
        Value: 0
PublicLoadBalancerSG:
  Type: 'AWS::EC2::SecurityGroup'
  Properties:
    GroupDescription: Access to the public facing load balancer
    VpcId: !Ref VpcId
    SecurityGroupIngress:
      - CidrIp: 0.0.0.0/0
        IpProtocol: -1
PublicLoadBalancer:
  Type: 'AWS::ElasticLoadBalancingV2::LoadBalancer'
  Properties:
    Scheme: internet-facing
    LoadBalancerAttributes:
      - Key: idle_timeout.timeout_seconds
        Value: '30'
```

```
    Subnets: !Ref PublicSubnetIds
    SecurityGroups:
      - !Ref PublicLoadBalancerSG
PublicLoadBalancerListener:
  Type: 'AWS::ElasticLoadBalancingV2::Listener'
  Properties:
    DefaultActions:
      - Type: forward
        ForwardConfig:
          TargetGroups:
            - TargetGroupArn: !Ref ServiceTargetGroup
              Weight: 100
    LoadBalancerArn: !Ref PublicLoadBalancer
    Port: 80
    Protocol: HTTP
ServiceIngressfromLoadBalancer:
  Type: 'AWS::EC2::SecurityGroupIngress'
  Properties:
    Description: Ingress from the public ALB
    GroupId: !Ref ServiceSecurityGroup
    IpProtocol: -1
    SourceSecurityGroupId: !Ref PublicLoadBalancerSG
LogGroup:
  Type: 'AWS::Logs::LogGroup'
```

Rôles IAM pour Amazon ECS

Vous pouvez utiliser des CloudFormation modèles pour créer des rôles IAM à utiliser avec Amazon ECS. Pour plus d'informations sur les rôles IAM pour Amazon ECS, consultez la section Rôles IAM pour Amazon ECS.

Rôle d'exécution des tâches Amazon ECS

Le rôle d'exécution des tâches autorise le conteneur Amazon ECS et les agents Fargate à AWS effectuer des appels d'API en votre nom. Le rôle est requis en fonction des besoins de votre tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Le modèle suivant peut être utilisé pour créer un rôle d'exécution de tâche simple qui utilise la politique `AmazonECSTaskExecutionRolePolicy` gérée.

JSON

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Description": "CloudFormation template for ECS Task Execution Role",
  "Resources": {
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Statement": [
            {
              "Effect": "Allow",
              "Principal": {
                "Service": ["ecs-tasks.amazonaws.com"]
              },
              "Action": ["sts:AssumeRole"],
              "Condition": {
                "ArnLike": {
                  "aws:SourceArn": {
                    "Fn::Sub": "arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*"
                  }
                },
                "StringEquals": {
                  "aws:SourceAccount": {
                    "Ref": "AWS::AccountId"
                  }
                }
              }
            }
          ]
        },
        "Path": "/",
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
      }
    }
  },
  "Outputs": {
    "ECSTaskExecutionRoleARN": {
      "Description": "ARN of the ECS Task Execution Role",
      "Value": {
        "Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]
```

```
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleARN"
        }
      }
    },
    "ECSTaskExecutionRoleName": {
      "Description": "Name of the ECS Task Execution Role",
      "Value": {
        "Ref": "ECSTaskExecutionRole"
      },
      "Export": {
        "Name": {
          "Fn::Sub": "${AWS::StackName}-ECSTaskExecutionRoleName"
        }
      }
    }
  }
}
```

YAML

```
AWSTemplateFormatVersion: '2010-09-09'
Description: 'CloudFormation template for ECS Task Execution Role'
Resources:
  ECSTaskExecutionRole:
    Type: AWS::IAM::Role
    Properties:
      AssumeRolePolicyDocument:
        Statement:
          - Effect: Allow
            Principal:
              Service: [ecs-tasks.amazonaws.com]
            Action: ['sts:AssumeRole']
            Condition:
              ArnLike:
                aws:SourceArn: !Sub arn:aws:ecs:${AWS::Region}:${AWS::AccountId}:*
              StringEquals:
                aws:SourceAccount: !Ref AWS::AccountId
      Path: /
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
```

```
Outputs:
  ECSTaskExecutionRoleARN:
    Description: ARN of the ECS Task Execution Role
    Value: !GetAtt ECSTaskExecutionRole.Arn
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleARN"
  ECSTaskExecutionRoleName:
    Description: Name of the ECS Task Execution Role
    Value: !Ref ECSTaskExecutionRole
    Export:
      Name: !Sub "${AWS::StackName}-ECSTaskExecutionRoleName"
```

Bonnes pratiques Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes pour le réseau Amazon ECS.

Aperçu des pratiques exemplaires	En savoir plus
Connexion d'applications à Internet	Connexion d'applications Amazon ECS à Internet
Recevez des connexions entrantes vers Amazon ECS depuis Internet.	Pratiques exemplaires en matière de réception de connexions entrantes vers Amazon ECS depuis Internet
Connect Amazon ECS à d'autres AWS services au sein d'un VPC	Meilleures pratiques pour connecter Amazon ECS à AWS des services depuis l'intérieur de votre VPC
Services réseau interconn ectés Comptes AWS et VPC	Meilleures pratiques pour la mise en réseau des services Amazon ECS sur Comptes AWS et VPC
Résolution des problèmes réseau	AWS services pour le dépannage du réseau Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes pour Fargate sur Amazon ECS.

Aperçu des pratiques exemplaires	En savoir plus
Sécurité Fargate	Pratiques exemplaires en matière de sécurité Fargate dans Amazon ECS
Considérations relatives à la sécurité Fargate	Considérations relatives à la sécurité Fargate pour Amazon ECS
Comportement d'extraction d'image de conteneur Linux sur Fargate	Comportement d'extraction d'image de conteneur Linux sur Fargate pour Amazon ECS
Comportement d'extraction d'image de conteneur relatif aux conteneurs Windows sur Fargate	Comportement d'extraction d'image de conteneur relatif aux conteneurs Windows sur Fargate pour Amazon ECS
Mise hors service de tâche Fargate	Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes en matière de définition des tâches.

Aperçu des pratiques exemplaires	En savoir plus
Images de conteneurs	Bonnes pratiques en matière d'images de conteneurs Amazon ECS
Taille de la tâche	Bonnes pratique en matière de taille des tâches Amazon ECS

Aperçu des pratiques exemplaires	En savoir plus	
Pratiques exemplaires en matière de volume	Options de stockage pour les tâches Amazon ECS	

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes en matière de clusters et de capacité.

Aperçu des pratiques exemplaires	En savoir plus	
Sécurité Fargate	Pratiques exemplaires en matière de sécurité Fargate dans Amazon ECS	
Considérations relatives à la sécurité d'instance de conteneur EC2	Considérations relatives à la sécurité des instances de conteneur Amazon EC2 pour Amazon ECS	
Cluster Auto Scaling	Optimisation de l'autoscaling de cluster Amazon ECS	
Exploitation à grande échelle	Exploitation d'Amazon ECS à grande échelle	
Autoscaling et gestion des capacités	Pratiques exemplaires en matière d'autoscaling et de gestion des capacités Amazon ECS	
Pratiques exemplaires en matière de sélection d'instances pour les instances gérées Amazon ECS	Pratiques exemplaires en matière de sélection d'instances pour les instances gérées Amazon ECS	

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes pour les tâches et les services.

Aperçu des pratiques exemplaires	En savoir plus
Optimisation du temps de lancement des tâches	Optimisation de l'heure de lancement des tâches Amazon ECS
Paramètres de service	Pratiques exemplaires relatives aux paramètres d'un service Amazon ECS
Optimisation des paramètres de surveillance de l'état de l'équilibreur de charge	Optimisation des paramètres de surveillance de l'état de l'équilibreur de charge pour Amazon ECS
Optimisation des paramètres de drainage de la connexion de l'équilibreur de charge	Optimisation des paramètres de drainage de la connexion de l'équilibreur de charge pour Amazon ECS
Optimisation de l'autoscaling du service	Optimisation de l'autoscaling d'un service Amazon ECS

Vous pouvez utiliser l'une des pages suivantes pour découvrir les pratiques exemplaires opérationnelles les plus importantes en matière de sécurité.

Aperçu des pratiques exemplaires	En savoir plus
Sécurité du réseau	Pratiques exemplaires en matière de sécurité réseau pour Amazon ECS

Aperçu des pratiques exemplaires	En savoir plus	
Sécurité des tâches et des conteneurs	Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS	

Architecture de votre solution pour Amazon ECS

Vous devez bâtir vos applications de manière à ce qu'elles puissent s'exécuter sur des conteneurs. Un conteneur est une unité standardisée de développement logiciel qui contient tout ce dont votre application logicielle a besoin pour être exécutée. Cela inclut le code, l'exécution, les outils système et les bibliothèques système pertinents. Les conteneurs sont créés à partir d'un modèle en lecture seule appelé image. Une image de conteneur est un artefact statique contenant votre application et ses dépendances. Les images sont généralement créées à partir d'un fichier Dockerfile. Un Dockerfile est un fichier en texte brut qui contient les instructions pour créer un conteneur. Après leur création, ces images sont stockées dans un registre, comme Amazon ECR, d'où elles peuvent être téléchargées.

Après avoir créé et stocké votre image, vous créez une définition de tâche Amazon ECS. Une définition de tâche est le plan de votre application. Il s'agit d'un fichier texte au format JSON qui décrit les paramètres et un ou plusieurs conteneurs qui forment votre application. Par exemple, vous pouvez l'utiliser pour spécifier l'image et les paramètres du système d'exploitation, les conteneurs à utiliser, les ports à ouvrir pour votre application et les volumes de données à utiliser avec les conteneurs dans la tâche. Les paramètres spécifiques disponibles pour votre définition de tâche dépendent des besoins de votre application spécifique.

Après avoir défini votre définition de tâche, vous la déployez sous forme de service ou de tâche sur votre cluster. Un cluster est un regroupement logique de tâches ou de services qui s'exécute sur l'infrastructure de capacité enregistrée dans un cluster.

Une tâche est l'instanciation d'une définition de tâche au sein d'un cluster. Vous pouvez exécuter une tâche autonome ou exécuter une tâche dans le cadre d'un service. Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément le nombre souhaité de tâches dans un cluster Amazon ECS. Le principe est le suivant : si l'une de vos tâches échoue ou s'arrête pour une raison quelconque, le planificateur de service d'Amazon ECS service lance une autre instance en fonction de votre définition de tâche. Il procède ainsi pour le remplacer et donc maintenir le nombre de tâches souhaité dans le service.

L'agent de conteneur s'exécute sur chaque instance de conteneur dans un cluster Amazon ECS. L'agent envoie à Amazon ECS des informations relatives aux tâches en cours d'exécution et à l'utilisation des ressources de vos conteneurs. Il démarre et arrête les tâches lorsqu'il reçoit une requête de la part d'Amazon ECS.

Une fois la tâche ou le service déployés, vous pouvez utiliser l'un des outils suivants pour surveiller votre déploiement et votre application :

- CloudWatch
- Surveillance d’exécution

Capacity

La capacité correspond à l’infrastructure sur laquelle vos conteneurs s’exécutent. Amazon ECS propose trois types d’infrastructure pour vos clusters :

- Instances gérées Amazon ECS : gère AWS entièrement les instances Amazon EC2 sous-jacentes exécutées sur votre compte, y compris le provisionnement, l'application de correctifs et la mise à l'échelle. Cette option offre un équilibre optimal entre performances, rentabilité et simplicité opérationnelle.
- Sans serveur (Fargate) : ne payez que pour les ressources utilisées par vos tâches sans gérer aucune infrastructure. Idéal pour les charges de travail variables et pour démarrer rapidement.
- Instances Amazon EC2 : vous gérez directement les instances Amazon EC2 sous-jacentes, y compris la sélection, la configuration et la maintenance des instances.

Utilisez les instances gérées Amazon ECS lorsque :

- vous recherchez la simplicité de Fargate avec un meilleur contrôle de l’infrastructure sous-jacente ;
- vous avez besoin de performances prévisibles et d’une optimisation des coûts ;
- Vous souhaitez AWS gérer l'infrastructure tout en préservant la flexibilité.

Utilisez Fargate lorsque :

- vous souhaitez vous concentrer sur votre application sans gérer l’infrastructure ;
- vous avez des charges de travail variables ou imprévisibles ;
- vous débutez avec les conteneurs et souhaitez l’option de déploiement la plus simple.

Utiliser des instances Amazon EC2 lorsque :

- vous avez besoin d'exigences matérielles spécifiques (accélération du GPU, calcul haute performance) ;
- vous avez besoin de réserves de capacité ou de types d'instances spécifiques ;
- vous avez besoin de fonctionnalités privilégiées ou d'AMI personnalisées ;

L'infrastructure est spécifiée lors de la création d'un cluster. Le type d'infrastructure est également spécifié lors de l'enregistrement d'une définition de tâche. La définition de la tâche désigne l'infrastructure sous le nom de « type de lancement ». Vous pouvez également faire appel à des fournisseurs de capacité.

Points de terminaison de service

Le point de terminaison de service est l'URL du point d'entrée Amazon ECS que vous utilisez pour vous connecter au service par programmation en utilisant IPv4 (Internet Protocol version 4) ou IPv6 (Internet Protocol version 6). Par défaut, les requêtes que vous effectuez pour vous connecter à Amazon ECS par programmation utilisent des points de terminaison de service qui prennent uniquement en charge les requêtes IPv4. Pour vous connecter par programmation à Amazon ECS à l'aide de requêtes IPv4 ou IPv6, vous pouvez utiliser un point de terminaison à double pile. Pour de plus amples informations sur les points de terminaison à double pile, consultez la section Utilisation des points de terminaison à double pile Amazon ECS.

Mise en réseau

AWS les ressources sont créées dans des sous-réseaux. Lorsque vous utilisez des instances EC2, Amazon ECS lance les instances dans le sous-réseau que vous spécifiez lorsque vous créez un cluster. Vos tâches s'exécutent dans le sous-réseau de l'instance. Pour Fargate ou les machines virtuelles sur site, vous spécifiez le sous-réseau lorsque vous exécutez une tâche ou créez un service.

Selon votre application, le sous-réseau peut être un sous-réseau privé ou public et le sous-réseau peut se trouver dans l'une des ressources suivantes : AWS

- Zones de disponibilité
- Local Zones
- Zones Wavelength
- Régions AWS

- AWS Outposts

Pour plus d'informations, consultez Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des Zones Wavelength ou Amazon Elastic Container Service sur AWS Outposts.

Vous pouvez connecter votre application à Internet à l'aide de l'une des méthodes suivantes :

- Un sous-réseau public avec une passerelle Internet

  Utilisez des sous-réseaux publics lorsque vous avez des applications publiques qui nécessitent de grandes quantités de bande passante ou une latence minimale. Les scénarios applicables incluent le streaming vidéo et les services de jeux.
- Un sous-réseau privé avec une passerelle NAT

  Utilisez des sous-réseaux privés lorsque vous souhaitez protéger vos conteneurs d'un accès externe direct. Les scénarios applicables incluent les systèmes de traitement des paiements ou les conteneurs stockant les données utilisateur et les mots de passe.
- AWS PrivateLink

  AWS PrivateLink Utilisez-le pour disposer d'une connectivité privée entre les VPC, les AWS services et vos réseaux locaux sans exposer votre trafic à l'Internet public.

Accès aux fonctionnalités

Vous pouvez utiliser les paramètres de votre compte Amazon ECS pour accéder aux fonctionnalités suivantes :

- Container Insights

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Les métriques incluent l'utilisation des ressources telles que l'UC, la mémoire, le disque et le réseau.
- Jonction `awsvpc`

  Pour certains types d'instances EC2, des interfaces réseau (ENI) supplémentaires peuvent être disponibles sur les instances de conteneur récemment lancées.
- Autorisation de balisage

Les utilisateurs doivent être autorisés à effectuer les actions qui créent la ressource, comme `ecsCreateCluster`. Si des balises sont spécifiées dans l'action de création de ressources, AWS exécute une autorisation supplémentaire sur l'`ecs:TagResource`action afin de vérifier si les utilisateurs ou les rôles sont autorisés à créer des balises.

- Conformité à Fargate FIPS-140

  Fargate prend en charge la norme fédérale de traitement de l'information (FIPS-140) qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent les informations sensibles. Il s'agit de la norme gouvernementale en vigueur aux États-Unis et au Canada, applicable aux systèmes qui doivent être conformes à la loi sur la gestion de la sécurité des informations fédérales (FISMA) ou au programme fédéral de gestion des risques et des autorisations (FedRAMP).

- Modification de l’heure de mise hors service de tâche Fargate

  Vous pouvez configurer la période d’attente avant que les tâches ne soient mises hors service pour l’application de correctif.

- VPC à double pile

  Autorisez les tâches à communiquer via l’IPv4, l’IPv6, ou via les deux protocoles.

- Format Amazon Resource Name (ARN)

  Certaines fonctionnalités, telles que l’autorisation d’étiquetage, nécessitent un nouveau format Amazon Resource Name (ARN).

Pour de plus amples informations, veuillez consulter Accès aux fonctionnalités d’Amazon ECS avec les paramètres du compte.

Rôles IAM

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d’autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Certaines fonctionnalités d’Amazon ECS nécessitent des rôles. Pour de plus amples informations, veuillez consulter Rôles IAM pour Amazon ECS.

Logging

La journalisation et la surveillance sont des aspects importants pour maintenir la fiabilité, la disponibilité et les performances des charges de travail Amazon ECS. Les options suivantes sont disponibles :

- Amazon CloudWatch logs : acheminez les journaux vers Amazon CloudWatch
- FireLenspour Amazon ECS : acheminez les journaux vers un AWS service ou une AWS Partner Network destination à des fins de stockage et d'analyse des journaux. AWS Partner Network Il s'agit d'une communauté mondiale de partenaires qui s'appuie sur des programmes, une expertise et des ressources pour créer, commercialiser et vendre des offres aux clients.

Pratiques exemplaires relatives aux images de conteneur

Les principes clés relatifs aux images de conteneurs Amazon ECS sont les suivants :

- Inclure toutes les dépendances dans l’image
- Exécuter un processus par conteneur
- Gérer `SIGTERM` pour des arrêts gracieux
- Écrire des journaux dans `stdout` et `stderr`
- Utilisez des balises uniques, évitez `latest` en production

Types de lancement et fournisseurs de capacité d'Amazon Amazon ECS

Amazon ECS propose deux méthodes pour configurer la capacité des charges de travail. Vous pouvez utiliser des types de lancement ou des fournisseurs de capacité. Les types de lancement incluent EC2, Fargate et External. Les fournisseurs de capacité offrent une flexibilité accrue et des fonctionnalités avancées pour la gestion des capacités. Vous pouvez exécuter des charges de travail sur un calcul sans serveur avec les fournisseurs de capacité Fargate et Fargate Spot, sur des instances EC2 autogérées via des fournisseurs de capacité de groupe Auto Scaling, ou sur un calcul entièrement géré à l'aide des fournisseurs de capacité Amazon ECS Managed Instances qui allient la simplicité de Fargate à la flexibilité du calcul EC2. Les fournisseurs de capacités offrent un meilleur contrôle sur l'allocation des ressources et peuvent contribuer à optimiser à la fois les performances

et les coûts. Les fournisseurs de capacité constituent la méthode recommandée pour configurer la capacité des charges de travail par rapport aux types de lancement traditionnels. Utilisez ce qui suit pour comprendre les différences entre les fournisseurs de capacité et les types de lancement.

Bonnes pratiques

Les meilleures pratiques sont les suivantes :

Utiliser les types de lancement pour définir la compatibilité de l'infrastructure

Les types de lancement définissent l'infrastructure sur laquelle les tâches et les services s'exécutent. Lorsque vous définissez des tâches, spécifiez `RequiresCompatibilities` d'inclure un ou plusieurs types de lancement compatibles avec les tâches. Vous pouvez utiliser les types de lancement suivants : EC2, Fargate, External et Amazon ECS Managed Instances. Bien que vous puissiez également utiliser le type de lancement pour exécuter vos tâches ou vos services, nous vous recommandons de l'utiliser uniquement pour définir les compatibilités dans vos définitions de tâches et d'utiliser des fournisseurs de capacité pour lancer des tâches ou des services. Notez que vous pouvez choisir un ou plusieurs types de lancement pour définir les compatibilités des tâches.

Utiliser des fournisseurs de capacité pour configurer la capacité de calcul

Lorsque vous lancez des tâches ou des services, configurez une stratégie de fournisseur de capacité. Amazon ECS prend en charge les fournisseurs de capacité suivants : Fargate et FARGATE_SPOT, les groupes Auto Scaling pour les instances EC2 autogérées et les instances gérées Amazon ECS. Notez que Spot Fleet n'est disponible qu'en tant que fournisseur de capacité et non en tant que type de lancement. Vous pouvez créer une ou plusieurs instances gérées Amazon ECS ou des fournisseurs de capacité de groupes Auto Scaling dans un cluster. Les fournisseurs de capacité Fargate et Fargate Spot sont créés et gérés par Amazon ECS sur chaque cluster et vous n'avez pas besoin de les créer. Un cluster peut combiner tous les types de fournisseurs de capacité, mais une stratégie de fournisseur de capacité ne peut pas combiner différents types de fournisseurs de capacité.

Mettre à jour la capacité des services

Vous pouvez simplement mettre à jour la stratégie d'un fournisseur de capacité pour un service afin de le déplacer d'un type de calcul à un autre.

Mutabilité des services

Amazon ECS prend en charge les services de mise à jour entre différents fournisseurs de capacité. Cela permet de :

- Mise à jour fluide des types de lancement aux fournisseurs de capacité
- Transitions entre les différents types de fournisseurs de capacité
- Tester différentes options de calcul sans recréer le service

Voici un aperçu général du processus :

1. Mettez à jour la définition de la tâche : assurez-vous d'`requiresCompatibilities`inclure le fournisseur de capacité cible, par exemple `MANAGED_INSTANCES`.

   > **Note**
   >
   > Les définitions de tâches doivent réussir la validation de compatibilité pour le fournisseur de capacité cible. Si la `requiresCompatibilities` vérification de la version de la définition de tâche échoue, l'`UpdateService`appel échoue.

2. Créer un fournisseur de capacité : si vous utilisez des groupes Amazon EC2 Auto Scaling personnalisés, créez le fournisseur de capacité.
3. Mettre à jour le service : modifiez le service pour utiliser une stratégie de fournisseur de capacité au lieu du type de lancement.
4. Validez le déploiement : vérifiez que les tâches se déploient correctement.
5. Surveillez et optimisez : ajustez les paramètres du fournisseur de capacité selon les besoins.

De fournisseur de capacité à fournisseur de capacité

Toutes les mises à jour de fournisseur de capacité à fournisseur de capacité sont prises en charge :

- Fournisseur de capacité de groupe Amazon EC2 Auto Scaling vers les instances gérées Amazon ECS
- Fournisseur de capacité Fargate pour les instances gérées Amazon ECS
- Du fournisseur de capacité de groupe Amazon EC2 Auto Scaling au fournisseur de capacité Fargate

- Instances gérées Amazon ECS vers le fournisseur de capacité Fargate
- Fournisseur de capacité Fargate au fournisseur de capacité de groupe Amazon EC2 Auto Scaling
- Instances gérées Amazon ECS vers le fournisseur de capacité de groupe Amazon EC2 Auto Scaling

Type de lancement vers le fournisseur de capacité

Toutes les mises à jour du type de lancement vers le fournisseur de capacité sont prises en charge :

- Type de lancement EC2 vers les instances gérées Amazon ECS
- Type de lancement Fargate vers les instances gérées Amazon ECS
- Type de lancement EC2 vers le fournisseur de capacité Fargate
- Type de lancement EC2 vers le fournisseur de capacité de groupe EC2 Auto Scaling
- Type de lancement Fargate vers le fournisseur de capacité de groupe Amazon EC2 Auto Scaling
- Type de lancement de Fargate vers le fournisseur de capacité Fargate
- Type de lancement externe vers les instances gérées Amazon ECS
- Type de lancement externe vers le fournisseur de capacité Fargate
- Type de lancement externe vers le fournisseur de capacité de groupe Amazon EC2 Auto Scaling

Type de lancement vers type de lancement

Les mises à jour du type de lancement vers le type de lancement ne sont pas prises en charge :

- Type de lancement EC2 vers type de lancement Fargate (utilisez plutôt le fournisseur de capacité Fargate)
- Fargate le type de lancement vers le type de lancement EC2 (utilisez plutôt le fournisseur de capacité de groupe Amazon EC2 Auto Scaling)

Au lieu de migrer d'un type de lancement à un autre, migrez vers le fournisseur de capacité équivalent pour bénéficier de fonctionnalités améliorées et d'une compatibilité future.

Type de fournisseur de capacité par rapport au type de lancement

Le fournisseur de capacité pour lancer les mises à jour de type n'est pas pris en charge. Si vous avez initialement créé un service avec un type de lancement, puis que vous l'avez mis à jour pour

utiliser des fournisseurs de capacité, vous pouvez rétablir le type de lancement initial du service. Pour revenir en arrière, transmettez une liste vide comme `capacityProviderStrategy` date d'appel`UpdateService`.

> **Note**
>
> Cela ramène uniquement le service au type de lancement spécifié lors de sa création initiale. Vous ne pouvez pas utiliser cette méthode pour passer d'un service à un autre type de lancement.

Utilisation des points de terminaison à double pile Amazon ECS

Les points de terminaison à double pile Amazon ECS prennent en charge les requêtes adressées à Amazon ECS via IPv4 (Internet Protocol version 4) et IPv6 (Internet Protocol version 6). Pour obtenir la liste des points de terminaison Amazon ECS, consultez la section Points de terminaison et quotas Amazon ECS dans le Références générales AWS.

Lorsque vous utilisez l'API REST, vous accédez directement à un point de terminaison Amazon ECS à l'aide du nom de point de terminaison (URI). Amazon ECS ne prend en charge que les noms de points de terminaison à double pile régionaux ; vous devez donc spécifier la région dans le nom.

Utilisez la convention de nommage suivante pour les noms des points de terminaison à double pile : `ecs.`*`region`*`.api.aws`.

Lorsque vous utilisez le AWS Command Line Interface (AWS CLI) et AWS les SDK, vous pouvez utiliser un paramètre ou un indicateur pour passer à un point de terminaison à double pile. Vous pouvez également spécifier directement le point de terminaison à double pile en remplaçant le point de terminaison Amazon ECS dans le fichier de configuration.

Les sections suivantes décrivent comment utiliser les points de terminaison à double pile à partir de l'API AWS CLI AWS REST, des SDK et de l'API.

Rubriques

- À l'aide de points de terminaison à double pile provenant du AWS CLI
- À l'aide de points de terminaison à double pile provenant du AWS Kits SDK
- Utilisation de points de terminaison Dual-Stack avec l'API REST

À l'aide de points de terminaison à double pile provenant du AWS CLI

Cette section fournit des exemples de AWS CLI commandes utilisées pour envoyer des requêtes à un point de terminaison à double pile. Pour plus d'informations sur l'installation AWS CLI ou la mise à jour vers la dernière version, consultez la section Installation ou mise à jour vers la AWS CLI dernière version du Guide de l'AWS Command Line Interface utilisateur de la version 2.

Pour utiliser un point de terminaison à double pile, vous pouvez définir la valeur `use_dualstack_endpoint` de configuration sur `true` dans le `config` fichier AWS CLI afin de diriger toutes les requêtes Amazon ECS effectuées par la `ecs` AWS CLI commande vers le point de terminaison à double pile pour la région spécifiée. Vous indiquez la Région dans le fichier `config` ou dans une commande à l'aide de l'option `--region`. Pour plus d'informations sur les fichiers de configuration pour le AWS CLI, consultez la section Paramètres des fichiers de configuration et d'identification AWS CLI dans le Guide de l'AWS Command Line Interface utilisateur de la version 2.

Si vous souhaitez utiliser un point de terminaison à double pile pour des AWS CLI commandes spécifiques, vous pouvez utiliser l'une des méthodes suivantes :

- Vous pouvez utiliser le point de terminaison à double pile par commande en définissant le paramètre `--endpoint-url` sur `https://ecs.`*`aws-region`*`.api.aws` ou sur `http://ecs.`*`aws-region`*`.api.aws` pour toute commande `ecs`.

  L'exemple de commande suivant répertorie tous les clusters disponibles et utilise le point de terminaison à double pile pour la requête.

  ```
  $ aws ecs list-clusters --endpoint-url https://ecs.aws-region.api.aws
  ```

- Vous pouvez créer des profils distincts dans votre AWS Config dossier. Par exemple, vous pouvez créer un profil qui définit `use_dualstack_endpoint` sur `true` et un profil qui ne définit pas `use_dualstack_endpoint`. Lorsque vous exécutez une commande, spécifiez le profil adéquat selon que vous comptez utiliser ou non le point de terminaison Dual-Stack.

À l'aide de points de terminaison à double pile provenant du AWS Kits SDK

Cette section fournit des exemples d'accès à un point de terminaison à double pile à l'aide des kits de AWS développement logiciel (SDK).

AWS SDK for Java 2.x

L'exemple suivant montre comment spécifier un point de terminaison à double pile pour la région `us-east-1` à l'aide de AWS SDK for Java 2.x.

```
Region region = Region.US_EAST_1
EcsClient client =
 EcsClient.builder().region(region).dualstackEnabled(true).build();
```

AWS SDK for Go

L'exemple suivant montre comment spécifier un point de terminaison à double pile pour la région `us-east-1` à l'aide de AWS SDK for Go.

```
sess := session.Must(session.NewSession())
svc := ecs.New(sess, &aws.Config{
    Region: aws.String(endpoints.UsEast1RegionID),
    Endpoint: aws.String("https://ecs.us-east-1.api.aws")
})
```

Pour plus d'informations, consultez la section relative Dual-stack aux points de terminaison FIPS dans le Guide de référence des AWS kits de développement logiciel et des outils.

Utilisation de points de terminaison Dual-Stack avec l'API REST

Lorsque vous utilisez l'API REST, vous pouvez accéder directement à un point de terminaison à double pile si vous le spécifiez dans votre requête. L'exemple suivant utilise le point de terminaison à double pile pour répertorier tous les clusters Amazon ECS de la région `us-east-1`.

```
POST / HTTP/1.1
Host: ecs.us-east-1.api.aws
Accept-Encoding: identity
Content-Length: 2
X-Amz-Target: AmazonEC2ContainerServiceV20141113.ListClusters
X-Amz-Date: 20150429T170621Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{}
```

Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des Zones Wavelength

Amazon ECS prend en charge les charges de travail qui utilisent des zones locales, des zones de longueur d'onde et AWS Outposts lorsque le traitement de données local ou à faible latence est requis.

- Vous pouvez utiliser les zones locales comme extension de et Région AWS pour placer des ressources à plusieurs endroits, plus proches de vos utilisateurs finaux.
- Vous pouvez utiliser les zones Wavelength pour créer des applications qui offrent des latences ultra-faibles aux appareils 5G et aux utilisateurs finaux. Wavength déploie des services de AWS calcul et de stockage standard à la périphérie des réseaux 5G des opérateurs de télécommunications.
- AWS Outposts apporte des modèles natifs Services AWS, d'infrastructure et d'exploitation à pratiquement tous les centres de données, espaces de colocation ou installations sur site.

> **Important**
>
> Amazon ECS sur les AWS Fargate charges de travail n'est pas pris en charge dans les zones locales, les zones de longueur d'onde ou activé pour le AWS Outposts moment.

Pour plus d'informations sur les différences entre les zones locales, les zones de longueur d'onde et AWS Outposts , consultez la section Comment dois-je déterminer quand utiliser AWS Wavength, AWS Local Zones ou AWS Outposts pour les applications nécessitant une faible latence ou un traitement de données local dans les AWS Wavelength FAQ.

Sous-réseaux partagés

Vous pouvez utiliser Partage de VPC pour partager des sous-réseaux avec d'autres comptes AWS au sein d'un même AWS Organizations.

Vous pouvez utiliser des VPC partagés pour EC2 en tenant compte des considérations suivantes :

- Le propriétaire du sous-réseau VPC doit partager un sous-réseau avec un compte participant avant que ce compte puisse y créer des ressources Amazon ECS.

- Vous ne pouvez pas utiliser le groupe de sécurité par défaut VPC pour vos instances de conteneur, car il appartient au propriétaire. De plus, les participants ne peuvent pas lancer d'instances avec des groupes de sécurité détenus par d'autres participants ou par le propriétaire.
- Dans un sous-réseau partagé, le participant et le propriétaire contrôlent séparément les groupes de sécurité au sein de leur compte respectif. Le propriétaire du sous-réseau peut voir ces groupes de sécurité créés par les participants, mais ne peut pas exécuter d'actions sur ceux-ci. Si le propriétaire du sous-réseau souhaite supprimer ou modifier ces groupes de sécurité, le participant qui a créé le groupe de sécurité doit effectuer l'action.
- Le propriétaire du VPC partagé ne peut pas afficher, mettre à jour ou supprimer un cluster créé par un participant dans le sous-réseau partagé. Cela s'ajoute aux ressources VPC auxquelles chaque compte a un accès différent. Pour plus d'informations, consultez Responsabilités et autorisations pour les propriétaires et les participants dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez utiliser des VPC partagés pour Fargate en tenant compte des considérations suivantes :

- Le propriétaire du sous-réseau VPC doit partager un sous-réseau avec un compte participant avant que ce compte puisse y créer des ressources Amazon ECS.
- Vous ne pouvez pas créer de service ni exécuter de tâche à l'aide du groupe de sécurité par défaut pour le VPC, car celui-ci appartient au propriétaire. De plus, les participants ne peuvent pas créer un service ou exécuter une tâche à l'aide de groupes de sécurité appartenant à d'autres participants ou au propriétaire.
- Dans un sous-réseau partagé, le participant et le propriétaire contrôlent séparément les groupes de sécurité au sein de leur compte respectif. Le propriétaire du sous-réseau peut voir ces groupes de sécurité créés par les participants, mais ne peut pas exécuter d'actions sur ceux-ci. Si le propriétaire du sous-réseau souhaite supprimer ou modifier ces groupes de sécurité, le participant qui a créé le groupe de sécurité doit effectuer l'action.
- Le propriétaire du VPC partagé ne peut pas afficher, mettre à jour ou supprimer un cluster créé par un participant dans le sous-réseau partagé. Cela s'ajoute aux ressources VPC auxquelles chaque compte a un accès différent. Pour plus d'informations, consultez Responsabilités et autorisations pour les propriétaires et les participants dans le Guide de l'utilisateur Amazon VPC.

Pour plus d'informations sur le partage de sous-réseaux VPC, consultez Partager votre VPC avec d'autres comptes dans le Guide de l'utilisateur Amazon VPC.

Local Zones

Une zone locale est une extension d'une zone située Région AWS à proximité géographique de vos utilisateurs. Les Local Zones ont leurs propres connexions à Internet et prennent en charge Direct Connect. Les ressources créées dans une zone locale peuvent servir les utilisateurs locaux avec des communications à faible latence. Pour plus d'informations, consultez AWS Local Zones.

Une zone locale est représentée par un code de région suivi d'un identifiant qui indique l'emplacement (par exemple, `us-west-2-lax-1a`).

Pour utiliser une Local Zone, vous devez d'abord vous inscrire à la zone. Une fois que vous avez choisi, vous devez créer un VPC Amazon et un sous-réseau dans la Local Zone.

Vous pouvez lancer des instances Amazon EC2, des serveurs de fichiers Amazon FSx et des Application Load Balancers à utiliser pour vos clusters et tâches Amazon ECS.

Pour plus d'informations, voir Qu'est-ce que AWS les zones locales ? dans le Guide de l'utilisateur des zones AWS locales.

Zones Wavelength

Vous pouvez utiliser AWS Wavelength pour créer des applications qui offrent une latence ultra-faible aux appareils mobiles et aux utilisateurs finaux. Wavength déploie des services de AWS calcul et de stockage standard à la périphérie des réseaux 5G des opérateurs de télécommunications. Vous pouvez étendre un Amazon Virtual Private Cloud à une ou plusieurs zones Wavelength. Vous pouvez ensuite utiliser AWS des ressources telles que les instances Amazon EC2 pour exécuter des applications qui nécessitent une latence très faible et une connexion Services AWS dans la région.

Une zone Wavelength est une zone isolée située à l'emplacement du transporteur où l'infrastructure Wavelength est déployée. Les zones de longueur d'onde sont liées à un Région AWS. Une zone Wavelength est une extension logique d'une région et est gérée par le plan de contrôle de la région.

Une zone Wavelength est représentée par un code de région suivi d'un identifiant qui indique la zone Wavelength (par exemple, `us-east-1-wl1-bos-wlz-1`).

Pour utiliser une zone Wavelength, vous devez d'abord vous inscrire à la zone. Une fois que vous avez choisi, vous devez créer un VPC Amazon et un sous-réseau dans la zone Wavelength. Ensuite, vous pouvez lancer dans la zone vos instances Amazon EC2 à utiliser pour vos clusters et tâches Amazon ECS.

Pour plus d'informations, consultez Mise en route avec AWS Wavelength dans le Guide du développeur AWS Wavelength .

Les zones de longueur d'onde ne sont pas disponibles dans toutes Régions AWS. Pour plus d'informations sur les régions qui prennent en charge les zones Wavelength, consultez Zones Wavelength disponibles dans le Guide du développeur AWS Wavelength .

Amazon Elastic Container Service sur AWS Outposts

AWS Outposts permet AWS des services, une infrastructure et des modèles d'exploitation natifs dans les installations sur site. Dans AWS Outposts les environnements, vous pouvez utiliser les mêmes AWS API, outils et infrastructures que ceux que vous utilisez dans AWS Cloud.

Amazon ECS on AWS Outposts est idéal pour les charges de travail à faible latence qui doivent être exécutées à proximité des données et des applications sur site.

Pour plus d'informations à ce sujet AWS Outposts, consultez le Guide de AWS Outposts l'utilisateur .

Considérations

Voici quelques considérations relatives à l'utilisation d'Amazon ECS sur AWS Outposts :

- Amazon Elastic Container Registry et Network Load Balancer s'exécutent dans la AWS région, et non sur AWS Outposts. Gestion des identités et des accès AWS Cela a pour effet d'accroître les latences entre ces services et les conteneurs.
- AWS Fargate n'est pas disponible sur AWS Outposts.

Les considérations relatives à la connectivité réseau sont les suivantes pour AWS Outposts :

- Si la connectivité réseau entre votre AWS région AWS Outposts et votre région est perdue, vos clusters continueront de fonctionner. Toutefois, vous ne pouvez pas créer de nouveaux clusters ni effectuer de nouvelles actions sur les clusters existants tant que la connectivité n'a pas été rétablie. En cas de défaillance d'instance, l'instance ne sera pas automatiquement remplacée. L'agent CloudWatch Logs ne sera pas en mesure de mettre à jour les journaux et les données d'événements.
- Nous vous recommandons de fournir une connectivité fiable, hautement disponible et à faible latence entre votre région AWS Outposts et sa AWS région.

Conditions préalables

Voici les prérequis à satisfaire pour utiliser Amazon ECS sur AWS Outposts :

- Vous devez avoir installé et configuré un Outpost dans votre centre de données sur site.
- Vous devez avoir une connexion réseau fiable entre votre Outpost et sa région AWS .

Vue d'ensemble de la création de clusters sur AWS Outposts

Voici un aperçu de la configuration :

1. Créez un rôle et une politique avec des droits actifs AWS Outposts.
2. Créez un profil d'instance IAM avec des droits sur AWS Outposts.
3. Créez un VPC ou utilisez-en un existant situé dans la même région que celle de vôtre AWS Outposts.
4. Créez un sous-réseau ou utilisez un sous-réseau existant associé à l' AWS Outposts.

   Il s'agit du sous-réseau sur lequel s'exécutent les instances de conteneur.
5. Créez un groupe de sécurité pour les instances de conteneur dans votre cluster.
6. Créez un nouveau cluster Amazon ECS.
7. Définissez les variables d'environnement de l'agent de conteneur Amazon ECS pour lancer l'instance dans le cluster.
8. Exécutez un conteneur.

Pour plus d'informations sur la manière d'intégrer Amazon ECS à AWS Outposts, consultez la section Étendre Amazon ECS sur deux AWS Outposts racks.

L'exemple suivant crée un cluster Amazon ECS sur un AWS Outposts.

1. Créez un rôle et une politique avec des droits actifs AWS Outposts.

   Le fichier `role-policy.json` est le document de politique qui contient l'effet et les actions des ressources. Pour plus d'informations sur le format de fichier, consultez PutRolePolicy la référence de l'API IAM

   ```
   aws iam create-role --role-name ecsRole \
   ```

```
    --assume-role-policy-document file://ecs-policy.json
aws iam put-role-policy --role-name ecsRole --policy-name ecsRolePolicy \
    --policy-document file://role-policy.json
```

2. Créez un profil d'instance IAM avec des droits sur AWS Outposts.

```
aws iam create-instance-profile --instance-profile-name outpost
aws iam add-role-to-instance-profile --instance-profile-name outpost \
    --role-name ecsRole
```

3. Créez un VPC.

```
aws ec2 create-vpc --cidr-block 10.0.0.0/16
```

4. Créez un sous-réseau associé à votre AWS Outposts.

```
aws ec2 create-subnet \
    --cidr-block 10.0.3.0/24 \
    --vpc-id vpc-xxxxxxxx \
    --outpost-arn arn:aws:outposts:us-west-2:123456789012:outpost/op-
xxxxxxxxxxxxxxxx \
    --availability-zone-id usw2-az1
```

5. Créez un groupe de sécurité pour les instances de conteneur, en spécifiant la plage CIDR appropriée pour AWS Outposts. (Cette étape est différente pour AWS Outposts.)

```
aws ec2 create-security-group --group-name MyOutpostSG
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 22 --cidr 10.0.3.0/24
aws ec2 authorize-security-group-ingress --group-name MyOutpostSG --protocol tcp \
    --port 80 --cidr 10.0.3.0/24
```

6. Créez le cluster.
7. Définissez les variables d'environnement de l'agent de conteneur Amazon ECS pour lancer l'instance dans le cluster créé à l'étape précédente et définissez les identifications que vous souhaitez ajouter afin d'identifier le cluster (par exemple `Outpost` pour indiquer que le cluster est destiné à un Outpost).

```
#! /bin/bash
cat << 'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_IMAGE_PULL_BEHAVIOR=prefer-cached
```

```
ECS_CONTAINER_INSTANCE_TAGS={“environment”: ”Outpost”}
EOF
```

> **Note**
>
> Pour éviter les retards causés par l'extraction d'images de conteneur d'Amazon ECR dans la région, utilisez des caches d'images. Pour ce faire, chaque fois qu'une tâche est exécutée, configurez l'agent Amazon ECS pour qu'il utilise par défaut l'image mise en cache sur l'instance elle-même en définissant `ECS_IMAGE_PULL_BEHAVIOR` sur `prefer-cached`.

8. Créez l'instance de conteneur, en spécifiant le VPC et le sous-réseau de l' AWS Outposts sur lequel cette instance doit s'exécuter et un type d'instance disponible sur l' AWS Outposts. (Cette étape est différente pour AWS Outposts.)

   Le fichier `userdata.txt` contient les données des utilisateurs que l'instance peut utiliser pour effectuer des tâches de configuration automatisées courantes et même exécuter des scripts après le démarrage de l'instance. Pour plus d’informations sur le fichier pour les appels API, consultez la section Exécuter des commandes sur votre instance Linux au démarrage dans le Guide de l’utilisateur Amazon EC2.

   ```
   aws ec2 run-instances --count 1 --image-id ami-xxxxxxxx --instance-type c5.large \
       --key-name aws-outpost-key --subnet-id subnet-xxxxxxxxxxxxxxxxx \
       --iam-instance-profile Name outpost --security-group-id sg-xxxxxx \
       --associate-public-ip-address --user-data file://userdata.txt
   ```

   > **Note**
   >
   > Cette commande sert également à ajouter des instances supplémentaires au cluster. Tous les conteneurs déployés dans le cluster sont placés sur cet AWS Outposts.

Bonnes pratiques en matière d’images de conteneurs Amazon ECS

Une image de conteneur est un ensemble d'instructions expliquant comment créer le conteneur. Une image de conteneur contient le code de votre application et toutes les dépendances dont le code d'application a besoin pour s'exécuter. Les dépendances des applications incluent les packages de code source sur lesquels repose le code de votre application, un environnement d'exécution de

langage pour les langages interprétés et les packages binaires sur lesquels repose votre code lié dynamiquement.

Suivez ces directives lors de la conception et de la création de vos images de conteneur :

- Complétez vos images de conteneur en stockant toutes les dépendances de l'application sous forme de fichiers statiques à l'intérieur de l'image de conteneur.

  Si vous modifiez quelque chose dans l'image de conteneur, créez une autre image de conteneur avec les modifications.
- Exécutez un seul processus de candidature dans un conteneur.

  La durée de vie du conteneur est aussi longue que le processus de l'application. Amazon ECS remplace les processus qui ont échoué et détermine où lancer le processus de remplacement. Une image complète rend le déploiement global plus résilient.
- Faites en sorte que votre application gère le signal `SIGTERM`.

  Lorsque Amazon ECS arrête une tâche, il envoie d'abord un signal SIGTERM à la tâche pour informer l'application qu'elle doit se terminer et s'arrêter. Amazon ECS envoie ensuite un message `SIGKILL`. Lorsque les applications ignorent le signal `SIGTERM`, le service Amazon ECS doit attendre avant d'envoyer le signal `SIGKILL` pour mettre fin au processus.

  Vous devez déterminer le temps nécessaire à votre application pour terminer son travail et vous assurer que vos applications gèrent le signal `SIGTERM`. Le traitement des signaux de l'application doit empêcher celle-ci d'accepter de nouvelles tâches et terminer celles en cours, ou enregistrer les tâches inachevées dans un espace de stockage externe lorsque leur exécution prend trop de temps.
- Configurez des applications conteneurisées pour écrire des journaux dans `stdout` et `stderr`.

  Le découplage de la gestion des journaux de votre code d'application vous offre la flexibilité nécessaire pour ajuster la gestion des journaux au niveau de l'infrastructure. La modification de votre système de journalisation en est un exemple. Au lieu de modifier vos services, de créer et de déployer une nouvelle image de conteneur, vous pouvez ajuster les paramètres.
- Utilisez des balises pour gérer les versions des images de vos conteneurs.

  Les images de conteneurs sont stockées dans un registre de conteneurs. Chaque image d'un registre est identifiée par une balise. Une balise est appelée `latest`. Cette balise est dirigée vers la dernière version de l'image du conteneur de l'application, comme `HEAD` dans un référentiel

git. Nous vous recommandons d'utiliser uniquement la balise `latest` à des fins de test. Il est recommandé de baliser les images des conteneurs avec une balise unique pour chaque compilation. Nous vous recommandons de baliser vos images en utilisant le git SHA pour la validation git qui a été utilisée pour créer l'image.

Vous n'avez pas besoin de créer une image de conteneur pour chaque validation. Toutefois, nous vous recommandons de créer une nouvelle image de conteneur chaque fois que vous publiez une validation de code spécifique dans l'environnement de production. Nous vous recommandons également de baliser l'image avec une balise correspondant à la validation git du code contenu dans l'image. Si vous avez balisé l'image avec la validation git, vous pouvez trouver plus rapidement la version du code exécutée par l'image.

Nous vous recommandons également d'activer les balises d'image immuables dans Amazon Elastic Container Registry. Avec ce paramètre, vous ne pouvez pas modifier l'image du conteneur vers laquelle pointe une balise. Au lieu de cela, Amazon ECR impose qu'une nouvelle image soit téléchargée vers une nouvelle balise. Pour plus d'informations, veuillez consulter Mutabilité d'une étiquette d'image dans le Guide de l'utilisateur Amazon ECR.

Lorsque vous concevez votre application pour qu'elle soit exécutée AWS Fargate, vous devez choisir entre déployer plusieurs conteneurs dans la même définition de tâche ou déployer des conteneurs séparément dans plusieurs définitions de tâches. Si les conditions suivantes sont requises, nous vous recommandons de déployer plusieurs conteneurs dans une définition de tâche unique :

- Vos conteneurs partagent le même cycle de vie (autrement dit, ils sont lancés et résiliés conjointement).
- Vos conteneurs doivent être exécutés sur le même hôte sous-jacent (autrement dit, un conteneur référence l'autre sur un port localhost).
- Vos conteneurs partagent des ressources.
- Vos conteneurs partagent des volumes de données.

Si ces conditions suivantes ne sont pas requises, nous vous recommandons de déployer des conteneurs distincts dans plusieurs définitions de tâche. Cela vous permet de mettre à l'échelle, d'allouer et de désallouer les conteneurs séparément.

Bonnes pratiques de mise en réseau d'Amazon ECS

Les applications modernes sont généralement construites à partir de plusieurs composants distribués qui communiquent entre eux. Par exemple, une application mobile ou Web peut communiquer avec un point de terminaison d'API, et l'API peut être alimentée par plusieurs microservices qui communiquent via Internet.

Pour plus d'informations sur les meilleures pratiques en matière de connexion des applications à Internet, consultez la section Connexion d'applications Amazon ECS à Internet.

Pour plus d'informations sur les meilleures pratiques en matière de réception de connexions entrantes vers Amazon ECS depuis Internet, consultez la section Pratiques exemplaires en matière de réception de connexions entrantes vers Amazon ECS depuis Internet.

Pour plus d'informations sur les meilleures pratiques pour connecter Amazon ECS à d'autres AWS services, consultezMeilleures pratiques pour connecter Amazon ECS à AWS des services depuis l'intérieur de votre VPC.

Pour plus d'informations sur les pratiques exemplaires en matière de connexion de services au sein d'un VPC, consultez la section Pratiques exemplaires en matière de connexion des services Amazon ECS dans un VPC.

Pour plus d'informations sur les meilleures pratiques en matière de services de mise en réseau entre VPC Comptes AWS et VPC, consultezMeilleures pratiques pour la mise en réseau des services Amazon ECS sur Comptes AWS et VPC.

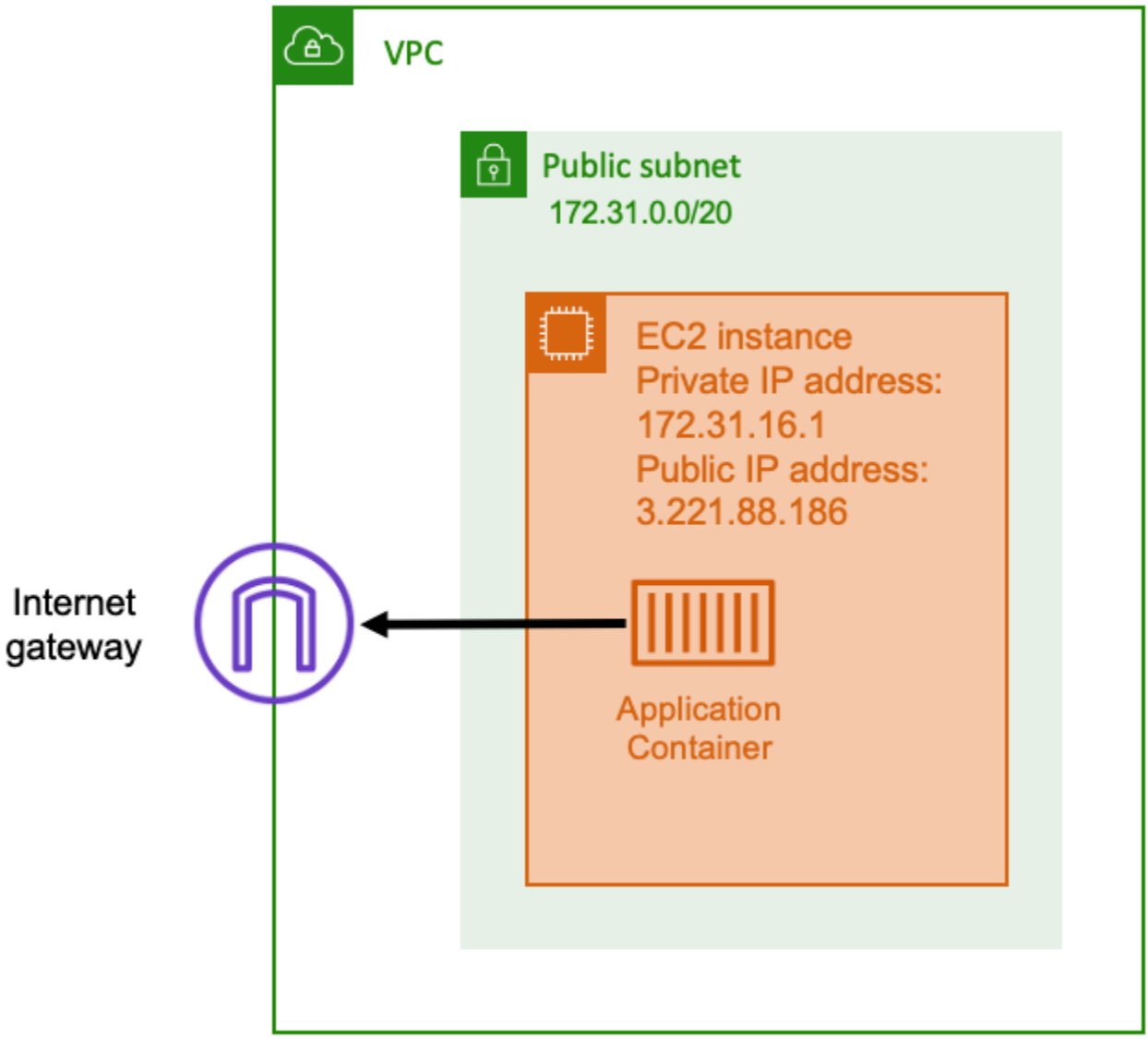
Pour plus d'informations sur les pratiques exemplaires en matière de services permettant de résoudre les problèmes de réseau, consultez la section AWS services pour le dépannage du réseau Amazon ECS.

Connexion d'applications Amazon ECS à Internet

La plupart des applications conteneurisées comportent au moins certains composants qui nécessitent un accès sortant à Internet. Par exemple, le dorsal d'une application mobile nécessite un accès sortant aux notifications push.

Amazon Virtual Private Cloud propose deux méthodes principales pour faciliter la communication entre votre VPC et Internet.

Sous-réseau public et passerelle Internet



Lorsque vous utilisez un sous-réseau public comportant une route vers une passerelle Internet, votre application conteneurisée peut être exécutée sur un hôte au sein d'un VPC sur un sous-réseau public. Une adresse IP publique est attribuée à l'hôte qui exécute votre conteneur. Cette adresse IP publique est routable depuis Internet. Pour plus d'informations, consultez Passerelles Internet dans le Guide de l'utilisateur Amazon VPC.

Cette architecture réseau facilite la communication directe entre l'hôte qui exécute votre application et les autres hôtes sur Internet. La communication est bidirectionnelle. Cela signifie que non seulement vous pouvez établir une connexion sortante avec n'importe quel autre hôte sur Internet, mais que d'autres hôtes sur Internet peuvent également tenter de se connecter à votre hôte. Par conséquent,

vous devez porter une attention particulière à votre groupe de sécurité et à vos règles de pare-feu. Ceci afin de garantir que d'autres hôtes sur Internet ne peuvent pas ouvrir de connexions que vous ne souhaitez pas voir ouvertes.

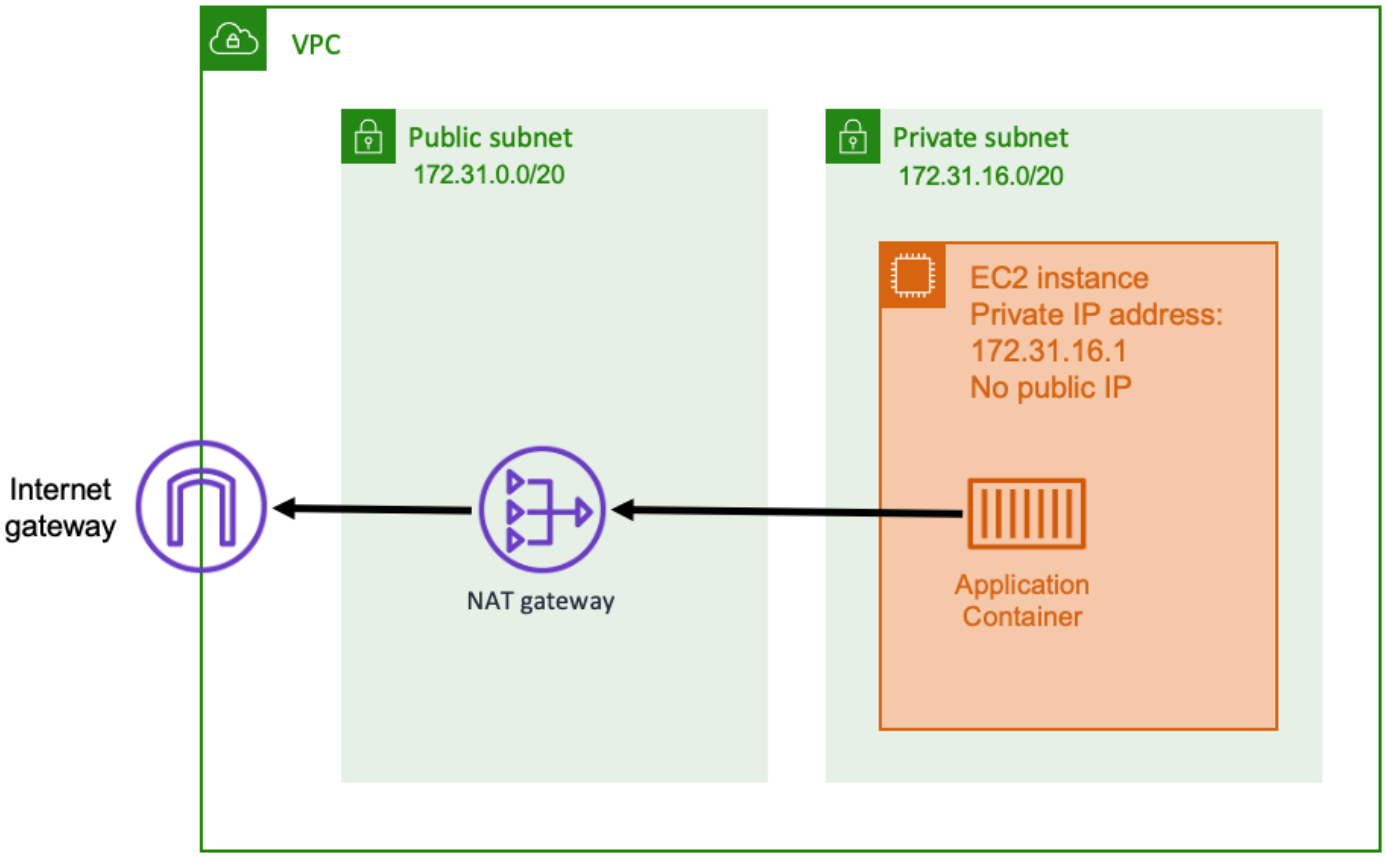
Par exemple, si votre application s'exécute sur Amazon EC2, assurez-vous que le port 22 pour l'accès SSH n'est pas ouvert. Dans le cas contraire, votre instance pourrait recevoir des tentatives de connexion SSH constantes de la part de robots malveillants sur Internet. Ces robots parcourent les adresses IP publiques. Une fois qu'ils ont trouvé un port SSH ouvert, ils tentent de forcer les mots de passe pour essayer d'accéder à votre instance. De ce fait, de nombreuses entreprises limitent l'utilisation des sous-réseaux publics et préfèrent que la plupart, sinon la totalité, de leurs ressources se trouvent dans des sous-réseaux privés.

L'utilisation de sous-réseaux publics pour la mise en réseau convient aux applications publiques qui nécessitent une bande passante importante ou une latence minimale. Les cas d'utilisation applicables comprennent les services de streaming vidéo et de jeux vidéo.

Cette approche réseau est prise en charge à la fois lorsque vous utilisez Amazon ECS sur Amazon EC2 et lorsque vous l'utilisez sur AWS Fargate.

- Amazon EC2 : vous pouvez lancer des instances EC2 sur un sous-réseau public. Amazon ECS utilise ces instances EC2 comme capacité de cluster, et tous les conteneurs exécutés sur les instances peuvent utiliser l'adresse IP publique sous-jacente de l'hôte pour le réseau sortant. Cela s'applique à la fois aux modes réseau `host` et `bridge`. Toutefois, le mode réseau `awsvpc` ne fournit pas d'ENI de tâche avec des adresses IP publiques. Par conséquent, l'utilisation d'une passerelle Internet n'est pas possible.
- Fargate : lorsque vous créez votre service Amazon ECS, spécifiez des sous-réseaux publics pour la configuration réseau de votre service et utilisez l'option Attribuer une adresse IP publique. Chaque tâche Fargate est mise en réseau dans le sous-réseau public et possède sa propre adresse IP publique pour une communication directe avec Internet.

Sous-réseau privé et passerelle NAT



Lorsque vous utilisez un sous-réseau privé et une passerelle NAT, vous pouvez exécuter votre application conteneurisée sur un hôte situé dans un sous-réseau privé. Ainsi, cet hôte possède une adresse IP privée qui est routable au sein de votre VPC, mais qui n'est pas routable depuis Internet. Cela signifie que les autres hôtes du VPC peuvent se connecter à l'hôte à l'aide de son adresse IP privée, mais que les autres hôtes sur Internet ne peuvent pas établir de communications entrantes avec l'hôte.

Avec un sous-réseau privé, vous pouvez utiliser une passerelle de traduction d'adresses réseau (NAT) pour autoriser un hôte d'un sous-réseau privé à se connecter à Internet. Les hôtes sur Internet reçoivent une connexion entrante qui semble provenir de l'adresse IP publique de la passerelle NAT située dans un sous-réseau public. La passerelle NAT est chargée de servir de pont entre Internet et le sous-réseau privé. Cette configuration est souvent préférée pour des raisons de sécurité, car elle signifie que votre VPC est protégé contre tout accès direct par des attaquants sur Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.

Cette approche de réseau privé convient aux scénarios dans lesquels vous souhaitez protéger vos conteneurs d'un accès externe direct. Les scénarios applicables incluent les systèmes de traitement des paiements ou les conteneurs stockant les données utilisateur et les mots de passe. La création et l'utilisation d'une passerelle NAT vous sont facturées dans votre compte. Les tarifs horaires d'utilisation de la passerelle NAT et de traitement des données s'appliquent également. Pour des raisons de redondance, vous devez disposer d'une passerelle NAT dans chaque zone de disponibilité. Ainsi, la perte de disponibilité d'une seule zone de disponibilité ne compromet pas votre connectivité sortante. De ce fait, si votre charge de travail est faible, il peut être plus rentable d'utiliser des sous-réseaux privés et des passerelles NAT.

Cette approche réseau est prise en charge à la fois lors de l'utilisation d'Amazon ECS sur Amazon EC2 et lors de son utilisation sur AWS Fargate.

- Amazon EC2 : vous pouvez lancer des instances EC2 sur un sous-réseau privé. Les conteneurs qui s'exécutent sur ces hôtes EC2 utilisent le réseau des hôtes sous-jacents, et les requêtes sortantes passent par la passerelle NAT.
- Fargate : lorsque vous créez votre service Amazon ECS, spécifiez des sous-réseaux privés pour la configuration réseau de votre service et n'utilisez pas l'option Attribuer une adresse IP publique. Chaque tâche Fargate est hébergée dans un sous-réseau privé. Son trafic sortant est acheminé via n'importe quelle passerelle NAT que vous avez associée à ce sous-réseau privé.

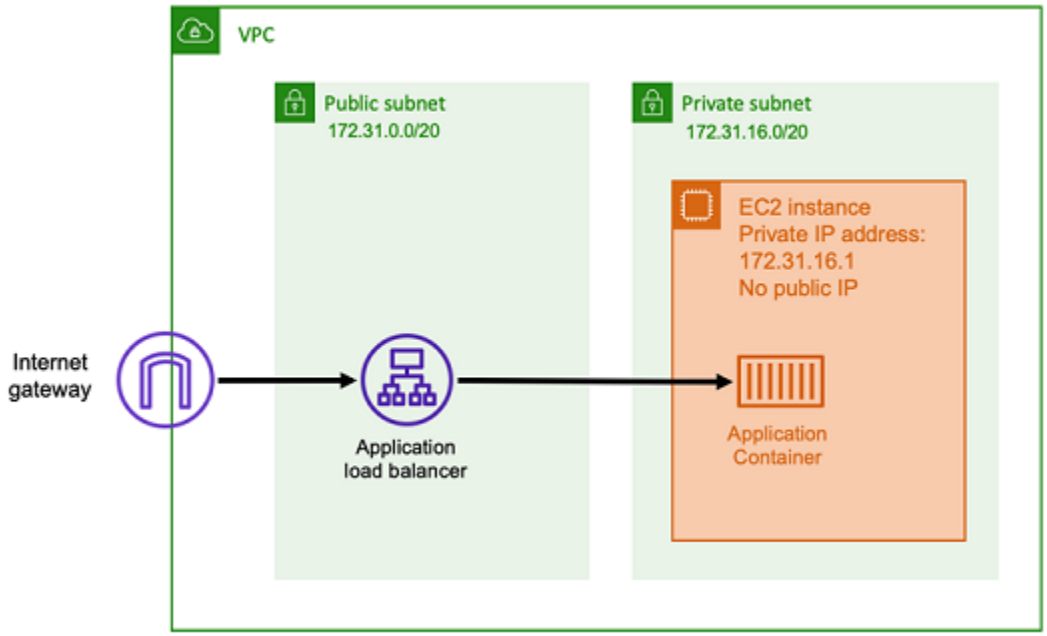
Pratiques exemplaires en matière de réception de connexions entrantes vers Amazon ECS depuis Internet

Si vous gérez un service public, vous devez accepter le trafic entrant en provenance d'Internet. Par exemple, votre site Web public doit accepter les requêtes HTTP entrantes provenant des navigateurs. Dans ce cas, les autres hôtes sur Internet doivent également établir une connexion entrante avec l'hôte de votre application.

Une solution à ce problème consiste à lancer vos conteneurs sur des hôtes situés dans un sous-réseau public doté d'une adresse IP publique. Cependant, nous ne recommandons pas cette solution pour les applications de grande envergure. Pour ces derniers, une meilleure approche consiste à disposer d'une couche d'entrée évolutive située entre Internet et votre application. Pour cette approche, vous pouvez utiliser n'importe lequel des services AWS répertoriés dans cette section comme entrée.

Application Load Balancer

Un Application Load Balancer fonctionne au niveau de la couche application. Il s'agit de la septième couche du modèle OSI (Open Systems Interconnection). Cela rend un Application Load Balancer adapté aux services HTTP publics. Si vous avez un site Web ou une API REST HTTP, un Application Load Balancer est un équilibreur de charge adapté à cette charge de travail. Pour plus d'informations, consultez la section Qu'est-ce qu'un Application Load Balancer ? dans le Guide de l'utilisateur des Application Load Balancer.



Avec cette architecture, vous créez un Application Load Balancer dans un sous-réseau public afin qu'il dispose d'une adresse IP publique et puisse recevoir des connexions entrantes depuis Internet. Lorsque l'Application Load Balancer reçoit une connexion entrante, ou plus précisément une requête HTTP, il ouvre une connexion à l'application à l'aide de son adresse IP privée. Ensuite, il transmet la requête via la connexion interne.

Un Application Load Balancer présente les avantages suivants :

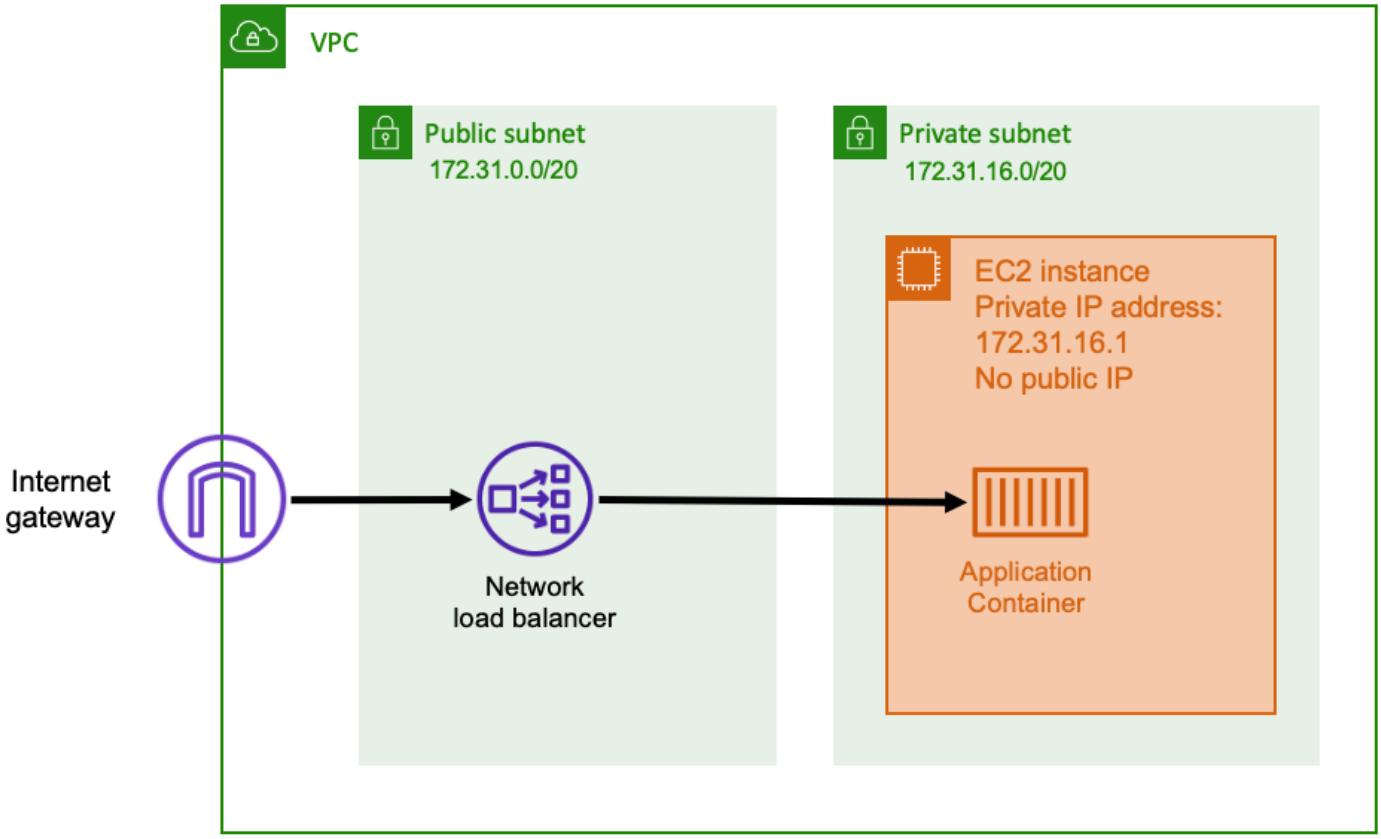
- SSL/TLS terminaison : un équilibreur de charge d'application peut gérer des communications HTTPS sécurisées et des certificats pour les communications avec les clients. Il peut éventuellement mettre fin à la connexion SSL au niveau de l'équilibreur de charge afin que vous n'ayez pas à gérer les certificats dans votre propre application.
- Routage avancé : un Application Load Balancer peut avoir plusieurs noms d'hôte DNS. Il dispose également de fonctionnalités de routage avancées pour envoyer des requêtes HTTP entrantes

vers différentes destinations en fonction de métriques telles que le nom d'hôte ou le chemin de la requête. Cela signifie que vous pouvez utiliser une seule Application Load Balancer comme entrée pour de nombreux services internes différents, voire des microservices sur différents chemins d'une API REST.

- Support du gRPC et Websockets : un Application Load Balancer peut gérer bien plus que du HTTP. Il peut également équilibrer la charge des services basés sur gRPC et Websocket, avec HTTP/2 support.
- Sécurité : un Application Load Balancer permet de protéger votre application contre le trafic malveillant. Il inclut des fonctionnalités telles que les mesures d'atténuation de la désynchronisation HTTP et est intégré au AWS Web Application Firewall (AWS WAF). AWS WAF peut filtrer davantage le trafic malveillant susceptible de contenir des modèles d'attaque, tels que l'injection SQL ou le cross-site scripting.

Network Load Balancer

Un Network Load Balancer fonctionne à la quatrième couche du modèle Open Systems Interconnection (OSI). Il convient aux protocoles non HTTP ou aux scénarios dans lesquels un chiffrement de bout en bout est nécessaire, mais ne possède pas les mêmes HTTP-specific fonctionnalités qu'un équilibreur de charge d'application. Par conséquent, un Network Load Balancer convient parfaitement aux applications qui n'utilisent pas le protocole HTTP. Pour plus d'informations, consultez la section Qu'est-ce qu'un Network Load Balancer ? dans le Guide de l'utilisateur des Network Load Balancer.



Lorsqu'un Network Load Balancer est utilisé comme entrée, il fonctionne de la même manière qu'un Application Load Balancer. Cela est dû au fait qu'il est créé dans un sous-réseau public et possède une adresse IP publique accessible sur Internet. Le Network Load Balancer ouvre ensuite une connexion à l'adresse IP privée de l'hôte qui exécute votre conteneur et envoie les paquets du côté public au côté privé.

Fonctionnalités du Network Load Balancer

Étant donné que le Network Load Balancer fonctionne à un niveau inférieur de la pile réseau, il ne possède pas le même ensemble de fonctionnalités que l'Application Load Balancer. Cependant, il présente les caractéristiques importantes suivantes.

- End-to-end chiffrement : étant donné qu'un équilibreur de charge réseau fonctionne au niveau de la quatrième couche du modèle OSI, il ne lit pas le contenu des paquets. Cela le rend adapté aux communications d'équilibrage de charge qui nécessitent un chiffrement de bout en bout.
- Chiffrement TLS : outre le chiffrement de bout en bout, Network Load Balancer peut également mettre fin aux connexions TLS. Ainsi, vos applications dorsales n'ont pas à implémenter leur propre protocole TLS.

- Support UDP : étant donné qu'un Network Load Balancer fonctionne au niveau de la quatrième couche du modèle OSI, il convient aux charges de travail non-HTTP et aux protocoles autres que le protocole TCP.

Fermeture des connexions

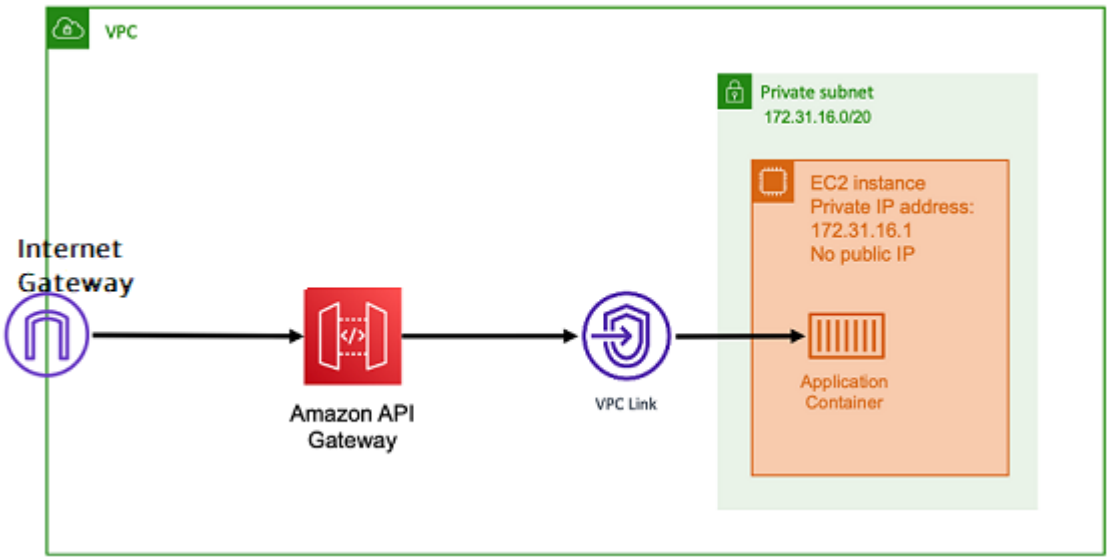
Comme le Network Load Balancer ne respecte pas le protocole d'application dans les couches supérieures du modèle OSI, il ne peut pas envoyer de messages de fermeture aux clients utilisant ces protocoles. Contrairement à l'Application Load Balancer, ces connexions doivent être fermées par l'application ou vous pouvez configurer le Network Load Balancer pour fermer les connexions de quatrième couche lorsqu'une tâche est arrêtée ou remplacée. Consultez les paramètres de résiliation de connexion pour les groupes cibles Network Load Balancer dans la Documentation Network Load Balancer.

Laisser le Network Load Balancer fermer les connexions au niveau de la quatrième couche peut entraîner l'affichage de messages d'erreur indésirables pour les clients, si ceux-ci ne les gèrent pas. Pour plus d'infirmations sur la configuration client recommandée, consultez la bibliothèque Builders ici.

Les méthodes de fermeture des connexions varient selon les applications, mais l'une d'entre elles consiste à s'assurer que le délai d'annulation de l'enregistrement de la cible du Network Load Balancer est plus long que le délai d'expiration de la connexion client. Le client devait d'abord expirer le délai imparti et se reconnecter progressivement à la tâche suivante via le Network Load Balancer, tandis que l'ancienne tâche épuisait lentement tous ses clients. Pour plus d'informations sur le délai d'annulation de l'enregistrement de la cible du Network Load Balancer, consultez la Documentation du Network Load Balancer.

API HTTP Amazon API Gateway

Amazon API Gateway convient aux applications HTTP présentant des pics soudains de volumes de requêtes ou de faibles volumes de requêtes. Pour plus d'informations, consultez la section Qu'est-ce qu'Amazon API Gateway ? dans le Guide du développeur API Gateway.



Le modèle de tarification pour Application Load Balancer et Network Load Balancer inclut un tarif horaire afin de maintenir les équilibreurs de charge disponibles pour accepter les connexions entrantes à tout moment. En revanche, API Gateway facture chaque requête séparément. Cela signifie que si aucune demande n'est reçue, il n'y a pas de frais. En cas de forte charge de trafic, un Application Load Balancer ou un Network Load Balancer peut traiter un plus grand volume de requêtes à un prix par requête inférieur à celui d'API Gateway. Cependant, si vous avez globalement peu de requêtes ou si vous connaissez des périodes de faible trafic, le prix cumulé pour l'utilisation de l'API Gateway devrait être plus rentable que de payer un tarif horaire pour maintenir un équilibreur de charge sous-utilisé. L'API Gateway peut également mettre en cache les réponses de l'API, ce qui peut entraîner une baisse des taux de requêtes du dorsal.

Fonctions API Gateway qui utilisent un lien VPC qui permet au service AWS géré de se connecter aux hôtes du sous-réseau privé de votre VPC, à l'aide de son adresse IP privée. Il peut détecter ces adresses IP privées en consultant les enregistrements de découverte de AWS Cloud Map services gérés par Amazon ECS Service Discovery.

API Gateway prend en charge les fonctionnalités ci-dessous.

- Le fonctionnement d'API Gateway est similaire à un équilibreur de charge, mais possède des fonctionnalités supplémentaires propres à la gestion des API
- L'API Gateway fournit des fonctionnalités supplémentaires en matière d'autorisation des clients, de niveaux d'utilisation et de request/response modification. Pour de plus amples informations, consultez la section Fonctionnalité d'Amazon API Gateway.

- L’API Gateway peut prendre en charge les points de terminaison de passerelle d’API périphériques, régionaux et privés. Les points de terminaison Edge sont disponibles via une CloudFront distribution gérée. Les points de terminaison régionaux et privés sont tous deux locaux à une région.
- SSL/TLS résiliation
- Routage de différents chemins HTTP vers différents microservices dorsaux

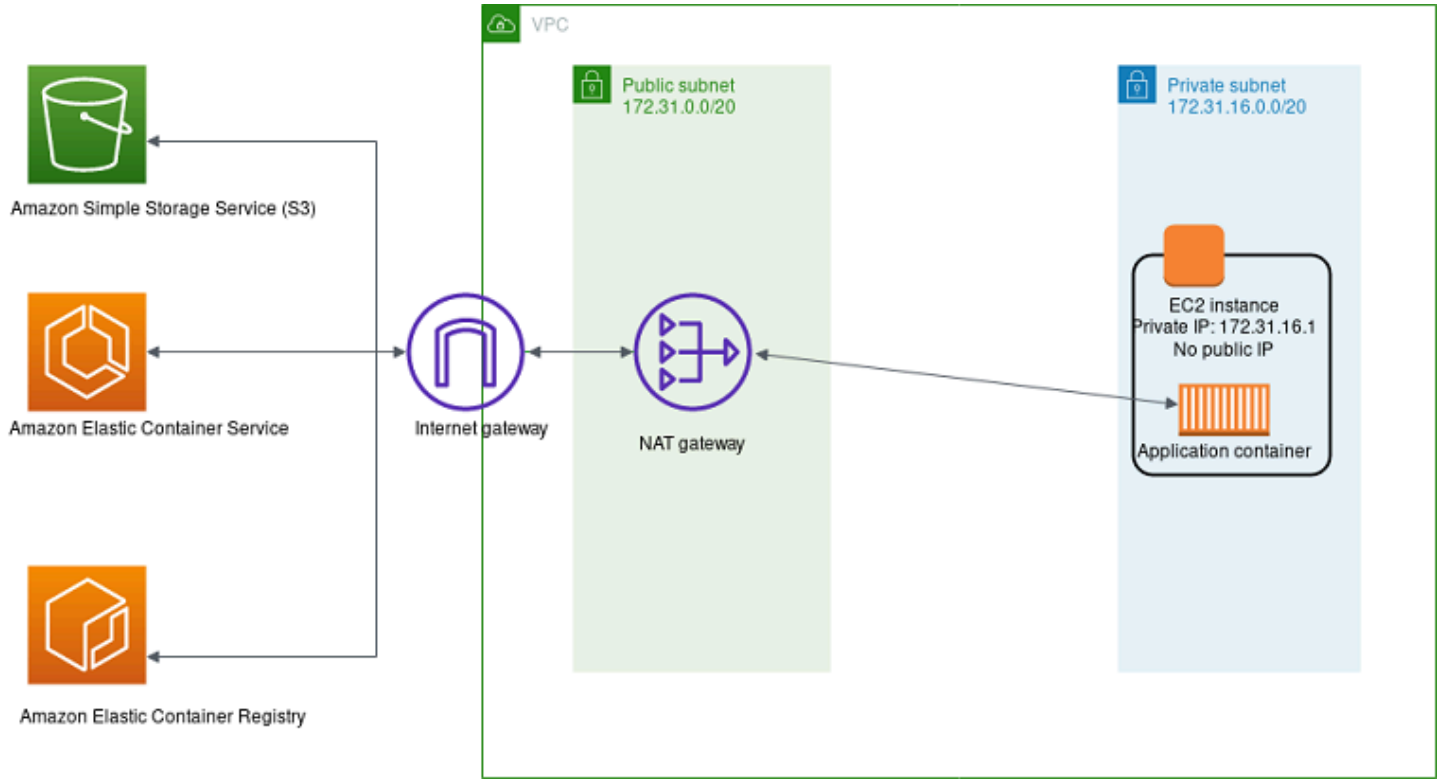
Outre les fonctionnalités précédentes, API Gateway prend également en charge l’utilisation d’autorisateurs Lambda personnalisés que vous pouvez utiliser pour protéger votre API contre toute utilisation non autorisée. Pour plus d'informations, consultez Notes de terrain : Container-based API sans serveur avec Amazon ECS et Amazon API Gateway.

Meilleures pratiques pour connecter Amazon ECS à AWS des services depuis l'intérieur de votre VPC

Pour qu’Amazon ECS fonctionne correctement, l’agent de conteneur Amazon ECS exécuté sur chaque hôte doit communiquer avec le plan de contrôle Amazon ECS. Si vous stockez vos images de conteneur dans Amazon ECR, les hôtes Amazon EC2 doivent communiquer avec le point de terminaison du service Amazon ECR et avec Amazon S3, où les couches d’images sont stockées. Si vous utilisez d'autres AWS services pour votre application conteneurisée, tels que les données persistantes stockées dans DynamoDB, vérifiez que ces services disposent également du support réseau nécessaire.

Passerelle NAT

L'utilisation d'une passerelle NAT est le moyen le plus simple de garantir que vos tâches Amazon ECS peuvent accéder à d'autres AWS services. Pour plus d’informations sur cette approche, consultez la section Sous-réseau privé et passerelle NAT.



L'utilisation de cette approche présente les inconvénients suivants :

- Vous ne pouvez pas limiter les destinations avec lesquelles la passerelle NAT peut communiquer. Vous ne pouvez pas non plus limiter les destinations vers lesquelles votre niveau dorsal peut communiquer sans interrompre toutes les communications sortantes de votre VPC.
- Les passerelles NAT facturent chaque Go de données qui transite par elle. Si vous utilisez la passerelle NAT pour l'une des opérations suivantes, chaque Go de bande passante vous est facturé :
  - Téléchargement de fichiers volumineux à partir d'Amazon S3
  - Exécution d'un volume élevé de requêtes de base de données vers DynamoDB
  - Extraction d'une image d'Amazon ECR

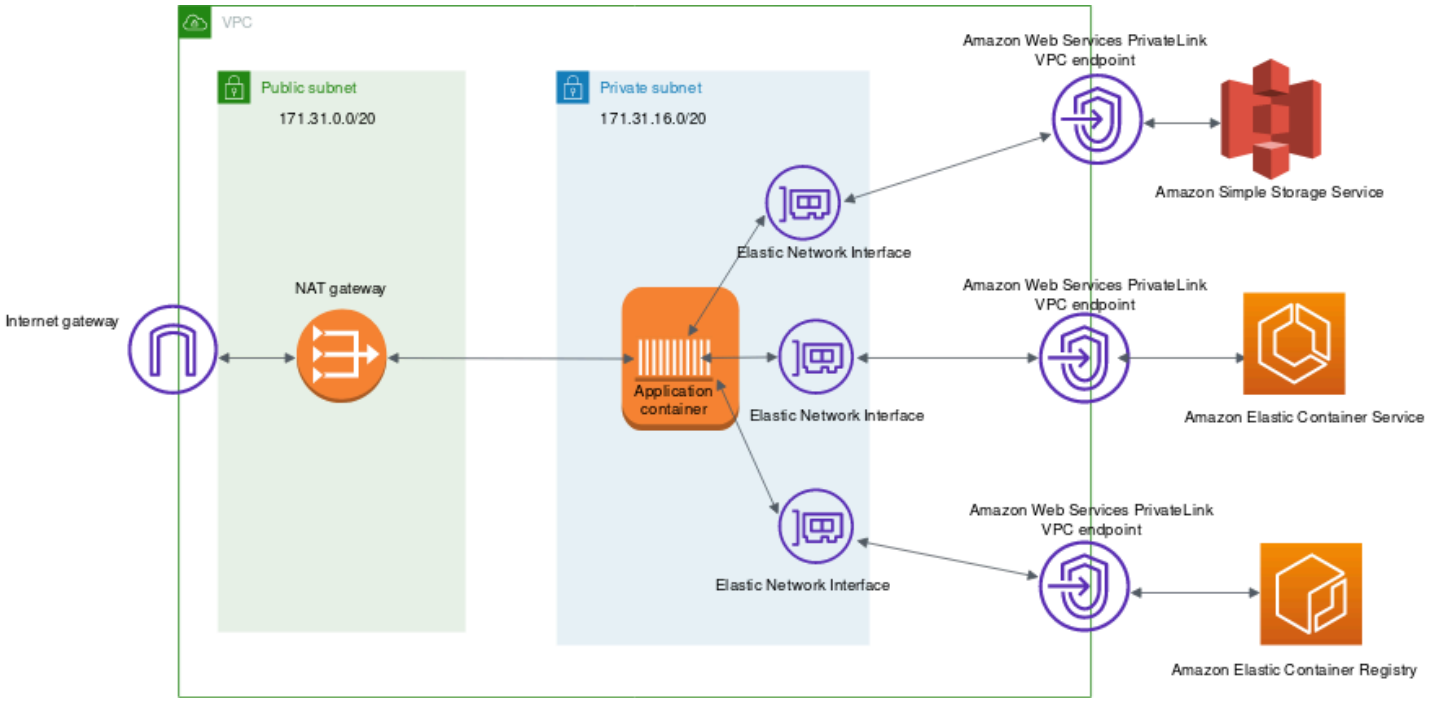
  De plus, les passerelles NAT prennent en charge une bande passante de 5 Gbit/s et s'adaptent automatiquement jusqu'à 45 Gbit/s. Si vous effectuez un routage via une seule passerelle NAT, les applications qui nécessitent des connexions à très haut débit peuvent se heurter à des contraintes réseau. Pour contourner le problème, vous pouvez répartir votre charge de travail sur plusieurs sous-réseaux et attribuer à chaque sous-réseau sa propre passerelle NAT.

AWS PrivateLink

AWS PrivateLink fournit une connectivité privée entre les VPC, AWS les services et vos réseaux locaux sans exposer votre trafic à l'Internet public.

Un point de terminaison de VPC permet des connexions privées entre votre VPC et les services AWS pris en charge, ainsi que les services de point de terminaison de VPC. Le trafic entre votre VPC et les autres services ne quitte pas le réseau Amazon. Un point de terminaison VPC ne nécessite pas de passerelle Internet, de passerelle privée virtuelle, de périphérique NAT, de connexion VPN ou de Direct Connect connexion. Les instances Amazon EC2 dans votre VPC ne nécessitent pas d'adresses IP publiques pour communiquer avec les ressources du service.

Le schéma suivant montre comment fonctionne la communication avec les AWS services lorsque vous utilisez des points de terminaison VPC au lieu d'une passerelle Internet. AWS PrivateLink fournit des interfaces réseau élastiques (ENI) à l'intérieur du sous-réseau, et les règles de routage VPC sont utilisées pour envoyer toute communication au nom d'hôte du service via l'ENI, directement au service de destination. AWS Ce trafic n'a plus besoin d'utiliser la passerelle NAT ou la passerelle Internet.



Voici quelques-uns des points de terminaison de VPC courants utilisés avec le service Amazon ECS.

- Points de terminaison de passerelle pour Amazon S3
- Point de terminaison de VPC DynamoDB

- Point de terminaison Amazon ECS
- Point de terminaison Amazon ECR

De nombreux autres AWS services prennent en charge les terminaux VPC. Si vous utilisez intensivement un service AWS , vous devez consulter la documentation spécifique à ce service et savoir comment créer un point de terminaison de VPC pour ce trafic.

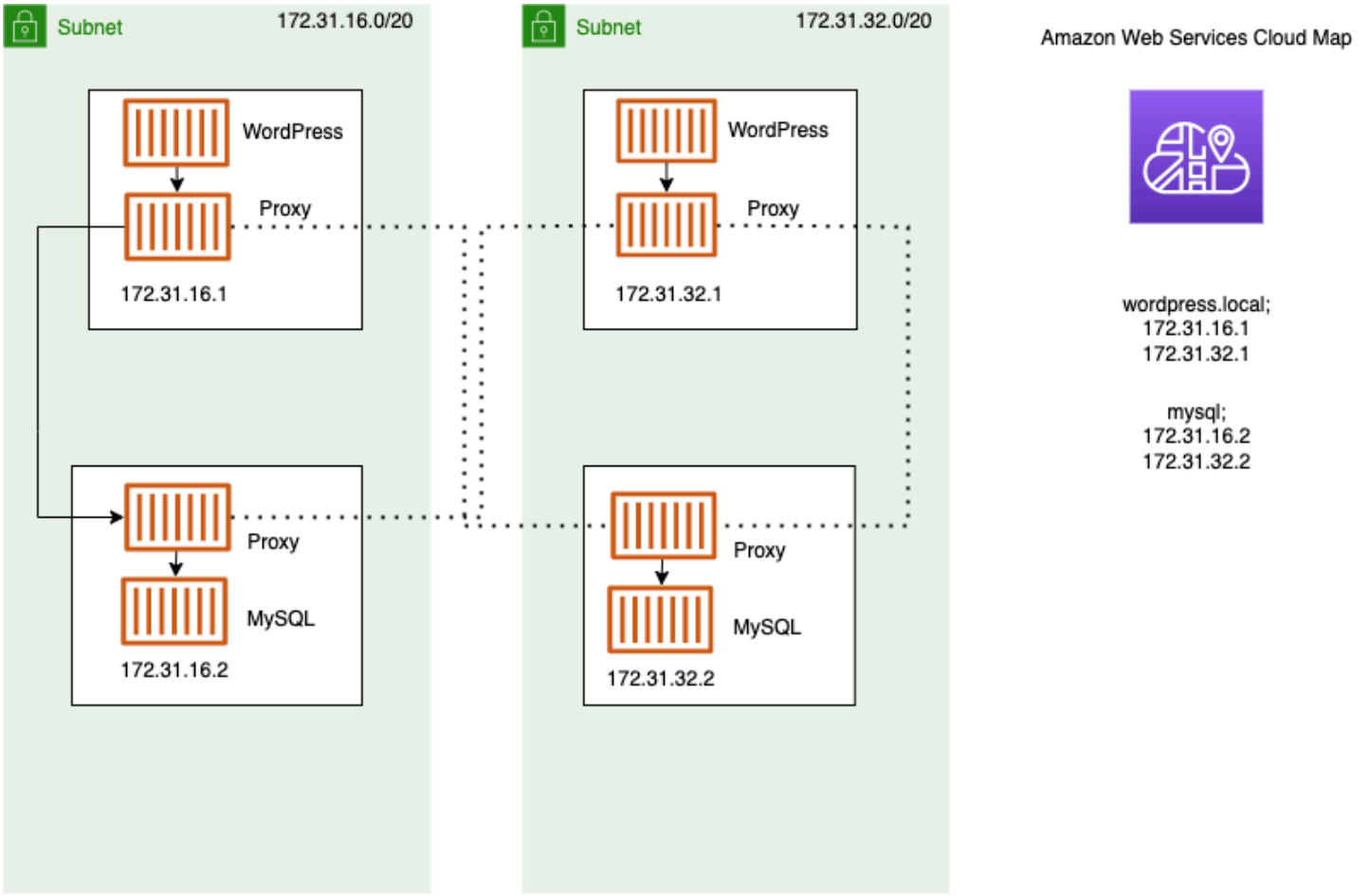
Pratiques exemplaires en matière de connexion des services Amazon ECS dans un VPC

À l'aide des tâches Amazon ECS dans un VPC, vous pouvez diviser les applications monolithiques en parties distinctes qui peuvent être déployées et mises à l'échelle indépendamment dans un environnement sécurisé. Cette architecture est appelée architecture orientée services (SOA) ou microservices. Cependant, il peut être difficile de s'assurer que toutes ces parties, à la fois à l'intérieur et à l'extérieur d'un VPC, peuvent communiquer entre elles. Il existe plusieurs approches pour faciliter la communication, qui présentent toutes des avantages et des inconvénients différents.

Utilisation de Service Connect

Nous recommandons Service Connect, qui fournit une configuration Amazon ECS pour la découverte de service, la connectivité et la surveillance du trafic. Avec Service Connect, vos applications peuvent utiliser des noms abrégés et des ports standard pour se connecter aux services d'un même cluster ou d'autres clusters, y compris entre des VPC d'une même région. Pour plus d'informations, consultez la section Amazon ECS Service Connect.

Lorsque vous utilisez Service Connect, Amazon ECS gère toutes les étapes de la découverte de services : création des noms susceptibles d'être découverts, gestion dynamique des entrées pour chaque tâche au démarrage et à l'arrêt des tâches, exécution d'un agent dans chaque tâche configuré pour découvrir les noms. Votre application peut rechercher les noms en utilisant les fonctionnalités standard pour les noms DNS et en établissant des connexions. Si votre application le fait déjà, vous n'avez pas besoin de modifier votre application pour utiliser Service Connect.



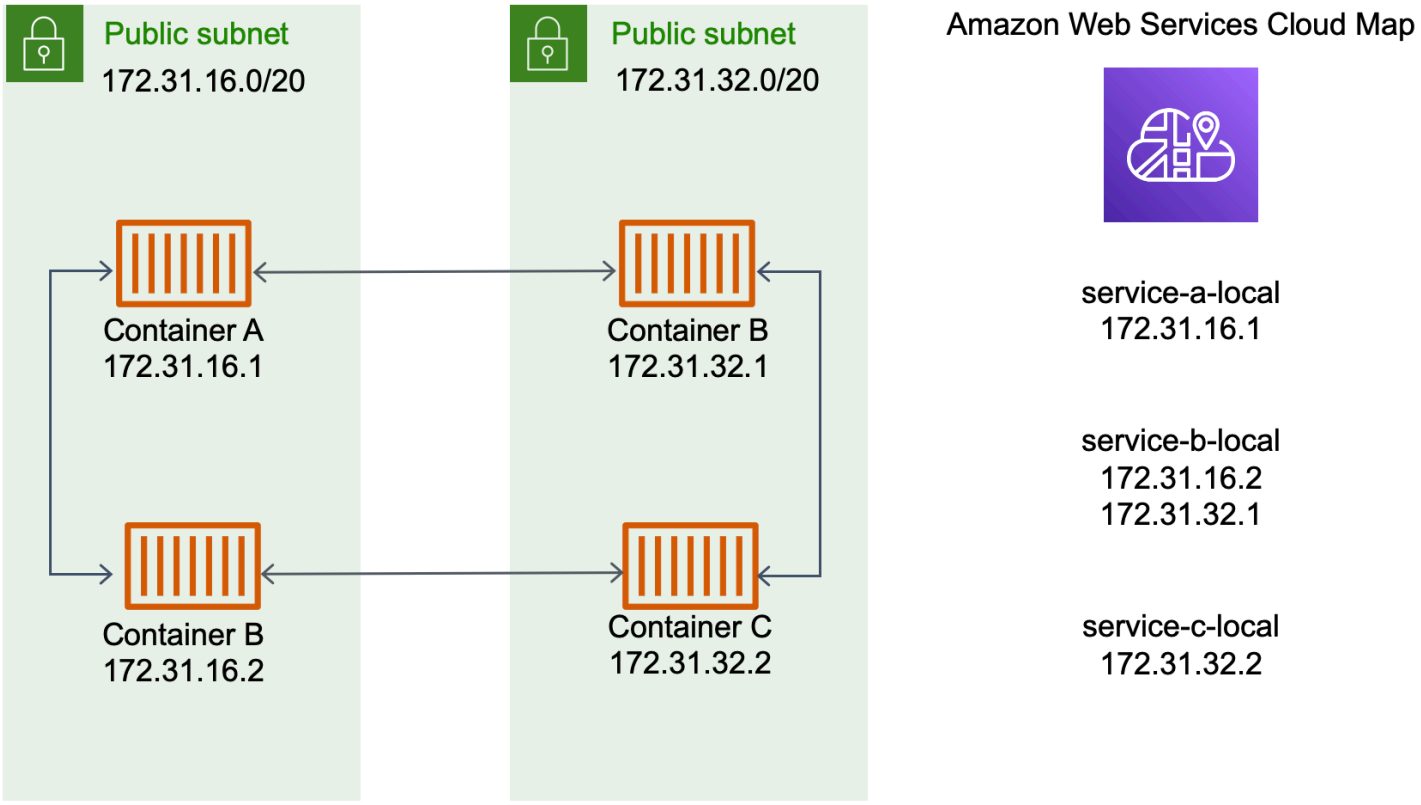
Les modifications ne se produisent que lors des déploiements

Vous fournissez la configuration complète dans chaque définition de service et de tâche. Amazon ECS gère les modifications apportées à cette configuration lors de chaque déploiement de service, afin de garantir que toutes les tâches d'un déploiement se comportent de la même manière. Par exemple, un problème courant avec le DNS en tant que découverte de service est le contrôle d'une migration. Si vous modifiez un nom DNS pour qu'il pointe vers les nouvelles adresses IP de remplacement, le délai TTL maximal peut être nécessaire avant que tous les clients commencent à utiliser le nouveau service. Avec Service Connect, le déploiement du client met à jour la configuration en remplaçant les tâches du client. Vous pouvez configurer le disjoncteur de déploiement et toute autre configuration de déploiement pour affecter les modifications de Service Connect de la même manière que pour tout autre déploiement.

Utilisation de la découverte de service

Une autre approche de la communication de service à service est la communication directe utilisant la découverte de service. Dans cette approche, vous pouvez utiliser l'intégration de la découverte

de AWS Cloud Map services avec Amazon ECS. À l'aide de la découverte de services, Amazon ECS synchronise la liste des tâches lancées avec AWS Cloud Map, ce qui permet de conserver un nom d'hôte DNS qui renvoie aux adresses IP internes d'une ou de plusieurs tâches provenant de ce service particulier. D'autres services au sein d'Amazon VPC peuvent utiliser ce nom d'hôte DNS pour envoyer le trafic directement vers un autre conteneur en utilisant son adresse IP interne. Pour de plus amples informations, consultez Découverte de service.



Dans le schéma précédent, il existe trois services. `service-a-local` possède un conteneur et communique avec `service-b-local`, qui possède deux conteneurs. `service-b-local` doit également communiquer avec `service-c-local`, qui possède un conteneur. Chaque conteneur de ces trois services peut utiliser les noms DNS internes de AWS Cloud Map pour rechercher les adresses IP internes d'un conteneur auprès du service en aval avec lequel il doit communiquer.

Cette approche de communication de service à service fournit une faible latence. À première vue, c'est également simple, car il n'y a aucun composant supplémentaire entre les conteneurs. Le trafic passe directement d'un conteneur à l'autre.

Cette approche convient lorsque vous utilisez le mode réseau `awsvpc`, où chaque tâche possède sa propre adresse IP unique. La plupart des logiciels ne prennent en charge que l'utilisation d'enregistrements DNS `A`, qui se résolvent directement en adresses IP. Lorsque vous utilisez le mode

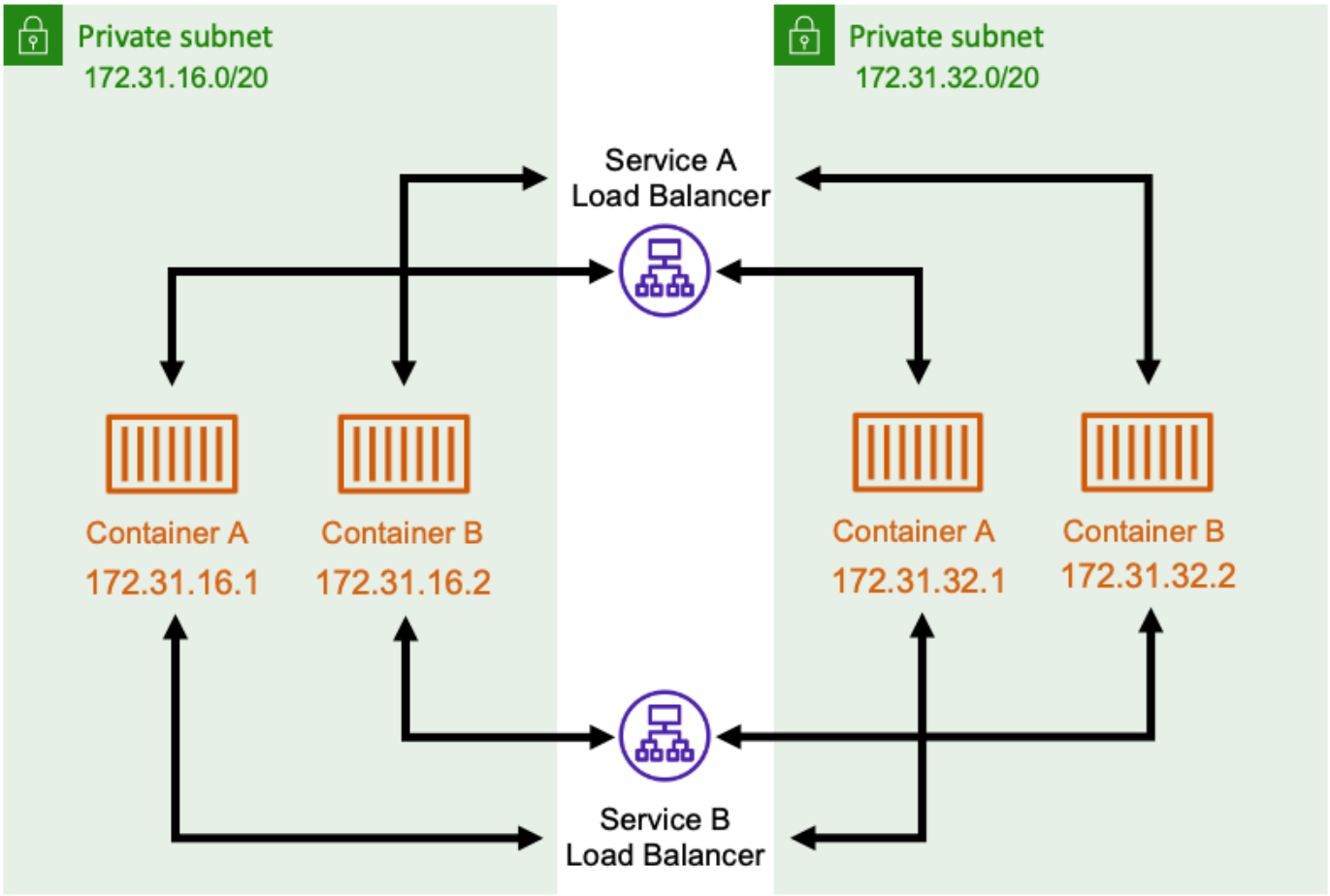
réseau `awsvpc`, l'adresse IP de chaque tâche est un enregistrement `A`. Toutefois, si vous utilisez le mode réseau `bridge`, plusieurs conteneurs peuvent partager la même adresse IP. En outre, les mappages de ports dynamiques entraînent l'attribution aléatoire de numéros de port aux conteneurs sur cette adresse IP unique. À ce stade, un enregistrement `A` ne suffit plus pour la découverte de services. Vous devez également utiliser un enregistrement `SRV`. Ce type d'enregistrement permet de suivre à la fois les adresses IP et les numéros de port, mais nécessite que vous configuriez les applications de manière appropriée. Certaines applications prédéfinies que vous utilisez peuvent ne pas prendre en charge les enregistrements `SRV`.

Un autre avantage du mode réseau `awsvpc` est que vous disposez d'un groupe de sécurité unique pour chaque service. Vous pouvez configurer ce groupe de sécurité pour autoriser les connexions entrantes provenant uniquement des services en amont spécifiques qui doivent communiquer avec ce service.

Le principal inconvénient de la communication directe entre services utilisant la découverte de service est que vous devez implémenter une logique supplémentaire pour effectuer de nouvelles tentatives et faire face aux échecs de connexion. Les enregistrements DNS ont une période de durée de vie (TTL) qui détermine combien de temps ils sont mis en cache. Il faut un certain temps pour que l'enregistrement DNS soit mis à jour et que le cache expire afin que vos applications puissent récupérer la dernière version de l'enregistrement DNS. Il se peut donc que votre application finisse par résoudre l'enregistrement DNS pour qu'il pointe vers un autre conteneur qui n'existe plus. Votre application doit gérer les nouvelles tentatives et avoir une logique lui permettant d'ignorer les mauvais dorsaux.

Utilisation d'un équilibreur de charge interne

Une autre approche de la communication entre services consiste à utiliser un équilibreur de charge interne. Un équilibreur de charge interne existe entièrement à l'intérieur de votre VPC et n'est accessible qu'aux services situés à l'intérieur de votre VPC.



L'équilibreur de charge maintient une haute disponibilité en déployant des ressources redondantes dans chaque sous-réseau. Lorsqu'un conteneur du `serviceA` doit communiquer avec un conteneur du `serviceB`, il ouvre une connexion avec l'équilibreur de charge. L'équilibreur de charge ouvre ensuite une connexion vers un conteneur du `service B`. L'équilibreur de charge sert de lieu centralisé pour gérer toutes les connexions entre chaque service.

Si un conteneur du `serviceB` s'arrête, l'équilibreur de charge peut le retirer du groupe. L'équilibreur de charge effectue également des surveillances de l'état par rapport à chaque cible en aval de son pool et peut automatiquement supprimer les mauvaises cibles du groupe jusqu'à ce qu'elles redeviennent saines. Les applications n'ont plus besoin de connaître le nombre de conteneurs en aval. Ils ouvrent simplement leurs connexions à l'équilibreur de charge.

Cette approche est avantageuse pour tous les modes de réseau. L'équilibreur de charge peut suivre les adresses IP des tâches en mode réseau `awsvpc`, ainsi que les combinaisons plus avancées d'adresses IP et de ports en mode réseau `bridge`. Il répartit le trafic de manière uniforme entre

toutes les combinaisons d'adresses IP et de ports, même si plusieurs conteneurs sont hébergés sur la même instance Amazon EC2, mais uniquement sur des ports différents.

Le seul inconvénient de cette approche est son coût. Pour être hautement disponible, l'équilibreur de charge doit disposer de ressources dans chaque zone de disponibilité. Cela entraîne des coûts supplémentaires en raison des frais généraux liés au paiement de l'équilibreur de charge et à la quantité de trafic qui passe par l'équilibreur de charge.

Cependant, vous pouvez réduire les frais généraux en faisant en sorte que plusieurs services partagent un équilibreur de charge. Cela convient particulièrement aux services REST qui utilisent un Application Load Balancer. Vous pouvez créer des règles de routage basées sur les chemins qui acheminent le trafic vers différents services. Par exemple, `/api/user/*` peut être acheminé vers un conteneur faisant partie du service `user`, alors que `/api/order/*` peut être acheminé vers le service `order` associé. Avec cette approche, vous ne payez que pour un Application Load Balancer et vous disposez d'une URL cohérente pour votre API. Cependant, vous pouvez répartir le trafic entre différents microservices sur le dorsal.

Meilleures pratiques pour la mise en réseau des services Amazon ECS sur Comptes AWS et VPC

Si vous faites partie d'une organisation composée de plusieurs équipes et divisions, vous déployez probablement des services indépendamment dans des VPC distincts au sein d'un VPC partagé Compte AWS ou dans des VPC associés à plusieurs personnes. Comptes AWS Quelle que soit la manière dont vous déployez vos services, nous vous recommandons de compléter vos composants réseau pour aider à acheminer le trafic entre les VPC. Pour cela, plusieurs AWS services peuvent être utilisés pour compléter vos composants réseau existants.

- AWS Transit Gateway — Vous devriez d'abord envisager ce service réseau. Ce service sert de hub central pour acheminer vos connexions entre Amazon VPC et Comptes AWS les réseaux locaux. Pour plus d'informations, consultez la section Qu'est-ce qu'une passerelle de transit ? dans le Guide des passerelles de transit Amazon VPC.
- Support Amazon VPC et VPN : vous pouvez utiliser ce service pour créer des connexions VPN de site à site afin de connecter des réseaux sur site à votre VPC. Pour plus d'informations, voir Qu'est-ce que c'est AWS Site-to-Site VPN ? dans le Guide de AWS Site-to-Site VPN l'utilisateur.
- Amazon VPC : vous pouvez utiliser l'appairage d'Amazon VPC pour vous aider à connecter plusieurs VPC, soit sur le même compte, soit sur plusieurs comptes. Pour plus d'informations, consultez Qu'est-ce que l'appairage de VPC ? dans le Guide d'appairage de VPC Amazon.

- VPC partagés : vous pouvez utiliser un VPC et des sous-réseaux VPC sur plusieurs comptes AWS . Pour de plus amples informations, consultez la section Utilisation de VPC partagés dans le Amazon VPC Guide de l'utilisateur.

AWS services pour le dépannage du réseau Amazon ECS

Les services et fonctionnalités suivants peuvent vous aider à mieux comprendre les configurations de votre réseau et de vos services. Vous pouvez utiliser ces informations pour résoudre les problèmes de mise en réseau et optimiser vos services.

CloudWatch Informations sur les conteneurs

CloudWatch Container Insights collecte, regroupe et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Les métriques incluent l'utilisation des ressources telles que l'UC, la mémoire, le disque et le réseau. Ils sont disponibles dans des tableaux CloudWatch de bord automatiques. Pour plus d'informations, consultez la section Configuration de Container Insights sur Amazon ECS dans le Guide de CloudWatch l'utilisateur Amazon.

AWS X-Ray

AWS X-Ray est un service de suivi que vous pouvez utiliser pour collecter des informations sur les requêtes réseau effectuées par votre application. Vous pouvez utiliser le SDK pour instrumenter votre application et capturer les horaires et les codes de réponse du trafic entre vos services, et entre vos services et les points de terminaison des services. AWS Pour plus d'informations, veuillez consulter la rubrique Présentation de AWS X-Ray dans le Guide du développeur AWS X-Ray .

Vous pouvez également explorer AWS X-Ray des graphiques illustrant la façon dont vos services interagissent les uns avec les autres. Vous pouvez également les utiliser pour explorer des statistiques agrégées sur les performances de chaque lien de service à service. Enfin, vous pouvez approfondir une transaction spécifique pour voir comment les segments représentant des appels réseau sont associés à cette transaction en particulier.

Vous pouvez utiliser ces fonctionnalités pour identifier s'il existe un goulot d'étranglement réseau ou si un service spécifique de votre réseau ne fonctionne pas comme prévu.

Journaux de flux VPC

Vous pouvez utiliser les journaux de flux Amazon VPC pour analyser les performances du réseau et résoudre les problèmes de connectivité. Lorsque les journaux de flux VPC sont activés, vous pouvez

capturer un journal de toutes les connexions de votre VPC. Il s'agit notamment des connexions aux interfaces réseau associées à Elastic Load Balancing, à Amazon RDS, aux passerelles NAT et à d'autres AWS services clés que vous pourriez utiliser. Pour plus d'informations, consultez Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Conseils d'optimisation du réseau

Il existe quelques paramètres que vous pouvez affiner afin d'optimiser votre réseau.

nofile ulimit

Si vous vous attendez à ce que votre application génère un trafic élevé et gère de nombreuses connexions simultanées, vous devez tenir compte du quota système pour le nombre de fichiers autorisés. Lorsqu'un grand nombre de sockets réseau sont ouverts, chacun doit être représenté par un descripteur de fichier. Si votre quota de descripteurs de fichiers est trop faible, cela limitera vos sockets réseau. Cela entraîne des échecs de connexion ou des erreurs. Vous pouvez mettre à jour le quota spécifique au conteneur pour le nombre de fichiers dans la définition de tâche Amazon ECS. Si vous utilisez Amazon EC2 (au lieu de AWS Fargate), vous devrez peut-être également ajuster ces quotas sur votre instance Amazon EC2 sous-jacente.

sysctl net

Une autre catégorie de paramètres réglables est celle des paramètres réseau `sysctl`. Vous devez vous référer aux paramètres spécifiques de la distribution Linux de votre choix. La plupart de ces paramètres ajustent la taille des tampons de lecture et d'écriture. Cela peut être utile dans certaines situations lors de l'exécution d'instances Amazon EC2 à grande échelle contenant de nombreux conteneurs.

Optimisation de l'heure de lancement des tâches Amazon ECS

Afin d'accélérer le lancement de vos tâches, tenez compte des recommandations suivantes.

- Mettre en cache les images de conteneurs et les instances binpack

  Si vous utilisez EC2, vous pouvez configurer le comportement d'extraction de l'agent de conteneur Amazon ECS pour `ECS_IMAGE_PULL_BEHAVIOR` : `prefer-cached`. L'image est extraite à distance s'il n'y a pas d'image mise en cache. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Le nettoyage d'image automatique est désactivé pour le conteneur afin de garantir que l'image mise en cache ne soit pas supprimée. Cela réduit le temps d'extraction

des images pour les lancements ultérieurs. L'effet de la mise en cache est encore plus important lorsque vous avez une densité de tâches élevée dans vos instances de conteneur, que vous pouvez configurer à l'aide de la stratégie de placement `binpack`. La mise en cache des images de conteneur est particulièrement avantageuse pour les charges de travail basées sur Windows qui ont généralement des tailles d'image de conteneur importantes (des dizaines de Go). Lorsque vous utilisez la stratégie de placement `binpack`, vous pouvez également envisager d'utiliser la jonction Elastic Network Interface (ENI) pour placer davantage de tâches en mode réseau `awsvpc` sur chaque instance de conteneur. La jonction ENI augmente le nombre de tâches que vous pouvez exécuter en mode `awsvpc`. Par exemple, une instance c5.large qui peut prendre en charge l'exécution simultanée de seulement 2 tâches peut exécuter jusqu'à 10 tâches avec la jonction ENI.

- Choix d'un mode réseau optimal

  Bien que le mode réseau `awsvpc` soit idéal dans de nombreux cas, ce mode réseau peut intrinsèquement augmenter la latence de lancement des tâches, car pour chaque tâche en mode `awsvpc`, les flux de travail Amazon ECS doivent fournir et associer une ENI en invoquant les API Amazon EC2, ce qui ajoute un surcoût de plusieurs secondes au lancement de vos tâches. En revanche, l'un des principaux avantages de l'utilisation du mode réseau `awsvpc` est que chaque tâche possède un groupe de sécurité qui autorise ou refuse le trafic. Cela signifie que vous disposez d'une plus grande flexibilité pour contrôler les communications entre les tâches et les services à un niveau plus détaillé. Si la vitesse de déploiement est votre priorité, vous pouvez envisager d'utiliser le mode `bridge` pour accélérer le lancement des tâches. Pour de plus amples informations, veuillez consulter the section called "Mode réseau AWSVPC".

- Suivez le cycle de vie de lancement de vos tâches pour identifier les opportunités d'optimisation

  Il est souvent difficile de savoir combien de temps prend le démarrage de votre application. Le lancement de votre image de conteneur, l'exécution de scripts de démarrage et d'autres configurations lors du démarrage de l'application peuvent prendre un temps surprenant. Vous pouvez utiliser le point de terminaison des métadonnées de tâche pour publier des métriques afin de suivre le temps de démarrage de l'application à partir de `StartedAt` dans la réponse des métadonnées du conteneur jusqu'à l'heure `StartedAt` de votre tâche ou service. Grâce à ces données, vous pouvez comprendre comment votre application contribue au temps de lancement total et identifier les domaines dans lesquels vous pouvez réduire les frais inutiles spécifiques à l'application et optimiser les images de vos conteneurs. Pour de plus amples informations, veuillez consulter Pratiques exemplaires en matière d'autoscaling et de gestion des capacités Amazon ECS.

- Choix d'un type d'instance optimal (pour EC2)

Le choix du type d'instance approprié dépend de la réserve de ressources (par exemple, UC, mémoire) que vous configurez pour votre tâche. Par conséquent, lors du dimensionnement de l'instance, vous pouvez calculer le nombre de tâches pouvant être placées sur une seule instance. Un exemple simple de tâche bien placée est l'hébergement de 4 tâches nécessitant 0,5 vCPU et 2 Go de réserves de mémoire dans une instance m5.large (prenant en charge 2 vCPU et 8 Go de mémoire). Les réserves de cette définition de tâche tirent pleinement parti des ressources de l'instance.

Exploitation d'Amazon ECS à grande échelle

Lorsque vous commencez à exploiter Amazon ECS à grande échelle, réfléchissez à la manière dont les quotas de service et les restrictions d'API pour Amazon ECS et ceux Services AWS qui s'intègrent à Amazon ECS peuvent vous affecter.

Rubriques

- Quotas de service Amazon ECS et seuils de limitation d'API
- Gestion des problèmes de limitation d'Amazon ECS

Quotas de service Amazon ECS et seuils de limitation d'API

Amazon ECS s'intègre à plusieurs d'entre eux Services AWS, notamment Elastic Load Balancing et Amazon EC2. AWS Cloud Map Grâce à cette intégration fine, Amazon ECS inclut plusieurs fonctionnalités telles que l'équilibrage de charge des services, la découverte de service, la mise en réseau des tâches et l'autoscaling de cluster. Amazon ECS et les autres solutions Services AWS qu'il intègre maintiennent tous les quotas de service et les limites de débit des API afin de garantir des performances et une utilisation constantes. Ces quotas de service empêchent également l'allocation accidentelle de ressources supérieures à celles nécessaires et protègent contre les actions malveillantes susceptibles d'augmenter votre facture.

En vous familiarisant avec vos quotas de service et les limites de débit des AWS API, vous pouvez planifier la mise à l'échelle de vos charges de travail sans vous soucier d'une dégradation inattendue des performances. Pour plus d'informations, consultez les sections Quotas de service Amazon ECS et Limitation des requêtes pour l'API Amazon ECS.

Lorsque vous effectuez une mise à l'échelle de vos charges de travail sur Amazon ECS, prenez en compte le quota de service suivant. Pour savoir comment demander une augmentation de quota

de service, consultez la section Gestion de votre Amazon ECS et de vos quotas de AWS Fargate service dans le Console de gestion AWS.

- AWS Fargate a des quotas qui limitent le nombre de tâches exécutées simultanément dans chacune d'elles Région AWS. Il existe des quotas à la fois pour les tâches On-Demand et pour les tâches Fargate Spot sur Amazon ECS. Chaque quota de service inclut également tous les pods Amazon EKS que vous utilisez sur Fargate. Pour de plus amples informations, consultez la section Quotas de service AWS Fargate dans le Guide de l'utilisateur Amazon Elastic Container Service pour AWS Fargate.
- Pour les tâches exécutées sur des instances Amazon EC2, le nombre maximal d'instances Amazon EC2 que vous pouvez enregistrer pour chaque cluster est de 5 000. Si vous utilisez l'autoscaling de cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, ou si vous gérez vous-même les instances Amazon EC2 pour votre cluster, ce quota peut devenir un obstacle au déploiement. Si vous avez besoin de plus de capacité, vous pouvez créer d'autres clusters ou demander une augmentation de quota de service.
- Si vous utilisez l'autoscaling de cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, tenez compte du quota `Tasks in the PROVISIONING state per cluster` lors du dimensionnement de vos services. Ce quota est le nombre maximum de tâches dans l'état `PROVISIONING` pour chaque cluster pour lesquelles les fournisseurs de capacité peuvent augmenter la capacité. Lorsque vous lancez un grand nombre de tâches simultanément, vous pouvez facilement atteindre ce quota. Par exemple, si vous déployez simultanément des dizaines de services, chacun comportant des centaines de tâches. Dans ce cas, le fournisseur de capacité doit lancer de nouvelles instances de conteneur pour placer les tâches lorsque la capacité du cluster est insuffisante. Pendant que le fournisseur de capacité lance des instances Amazon EC2 supplémentaires, le planificateur de service Amazon ECS continuera probablement à lancer des tâches en parallèle. Toutefois, cette activité peut être limitée en raison d'une capacité de cluster insuffisante. Le planificateur de service Amazon ECS met en œuvre une stratégie de ralentissement et de limitation exponentielle pour réessayer de placer des tâches lorsque de nouvelles instances de conteneur sont lancées. Par conséquent, les délais de déploiement ou de réduction horizontale peuvent être plus lents. Pour éviter cette situation, vous pouvez planifier vos déploiements de service de l'une des façons suivantes. Déployez un grand nombre de tâches qui ne nécessitent pas d'augmentation de la capacité du cluster, ou conservez une capacité de cluster disponible pour le lancement de nouvelles tâches.

En plus de prendre en compte le quota de service Amazon ECS lors de la mise à l'échelle de vos charges de travail, prenez également en compte le quota de service pour Services AWS les autres

services intégrés à Amazon ECS. La section suivante décrit en détail les principales limites de débit pour chaque service et fournit des recommandations pour faire face aux éventuels problèmes de limitation.

Elastic Load Balancing

Vous pouvez configurer vos services Amazon ECS pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches. Pour plus d'informations sur le choix d'un équilibreur de charge, consultez la section Utilisation de l'équilibrage de charge pour distribuer le trafic d'un service Amazon ECS.

Quotas de service Elastic Load Balancing

Lorsque vous mettez à l'échelle vos charges de travail, tenez compte des quotas du service Elastic Load Balancing suivants. La plupart des quotas de service Elastic Load Balancing sont ajustables, et vous pouvez demander une augmentation dans la console Service Quotas.

Application Load Balancer

Lorsque vous utilisez un Application Load Balancer, selon votre cas d'utilisation, vous devrez peut-être demander une augmentation de quota pour :

- Le quota `Targets per Application Load Balancer`, qui correspond au nombre de cibles derrière votre Application Load Balancer.
- Le quota `Targets per Target Group per Region`, qui correspond au nombre de cibles derrière vos groupes cibles.

Pour plus d'informations, consultez la section Quotas pour vos Application Load Balancer.

Network Load Balancer

Le nombre de cibles que vous pouvez enregistrer auprès d'un Network Load Balancer est soumis à des limites plus strictes. Lorsque vous utilisez un Network Load Balancer, vous souhaiterez souvent activer le support entre zones, ce qui implique des limites de mise à l'échelle supplémentaires sur les `Targets per Availability Zone Per Network Load Balancer`, qui correspond au nombre maximum de cibles par zone de disponibilité pour chaque Network Load Balancer. Pour plus d'informations, consultez la section Quotas pour vos Network Load Balancer.

Limitation de l'API Elastic Load Balancing

Lorsque vous configurez un service Amazon ECS pour utiliser un équilibreur de charge, les surveillances de l'état du groupe cible doivent réussir avant que le service ne soit considéré comme sain. Pour effectuer ces surveillances de l'état, Amazon ECS invoque les opérations de l'API Elastic Load Balancing en votre nom. Si vous avez configuré un grand nombre de services avec des équilibreurs de charge dans votre compte, vous risquez de constater un ralentissement des déploiements de service en raison d'une limitation potentielle spécifique aux opérations `RegisterTarget`, `DeregisterTarget` et `DescribeTargetHealth` de l'API Elastic Load Balancing. En cas de limitation, des erreurs de limitation apparaissent dans les messages d'événements de votre service Amazon ECS.

Si vous rencontrez une limitation des AWS Cloud Map API, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter les limites de limitation de vos AWS Cloud Map API. Pour plus d'informations sur la surveillance et le dépannage de telles erreurs de limitation, consultez la section Gestion des problèmes de limitation.

Interfaces réseau Elastic

Lorsque vos tâches utilisent le mode réseau `awsvpc`, Amazon ECS fournit une interface réseau Elastic (ENI) unique pour chaque tâche. Lorsque vos services Amazon ECS utilisent un équilibreur de charge Elastic Load Balancing, ces interfaces réseau sont également enregistrées en tant que cibles pour le groupe cible approprié défini dans le service.

Quotas de service de l'interface réseau Elastic

Lorsque vous exécutez des tâches utilisant le mode réseau `awsvpc`, une interface réseau Elastic unique est associée à chaque tâche. Si ces tâches doivent être atteintes via Internet, attribuez une adresse IP publique à l'interface réseau Elastic pour ces tâches. Lorsque vous mettez vos charges de travail Amazon ECS à l'échelle, tenez compte de ces deux quotas importants :

- Le `Network interfaces per Region` quota, qui correspond au nombre maximum d'interfaces réseau Région AWS pour votre compte.
- Le quota `Elastic IP addresses per Region`, qui correspond au nombre maximum d'adresses IP élastiques dans une Région AWS.

Ces deux quotas de service sont ajustables et vous pouvez demander une augmentation depuis votre console Service Quotas pour ces derniers. Pour plus d'informations, consultez la section Quotas de service Amazon VPC.

Pour les charges de travail Amazon ECS hébergées sur des instances Amazon EC2, lorsque vous exécutez des tâches en mode réseau `awsvpc`, tenez compte du quota de service `Maximum network interfaces`, qui correspond au nombre maximum d'instances réseau pour chaque instance Amazon EC2. Ce quota limite le nombre de tâches que vous pouvez placer sur une instance. Vous ne pouvez pas ajuster le quota et il n'est pas disponible dans la console Service Quotas. Pour de plus amples informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.

Bien que vous ne puissiez pas modifier le nombre d'interfaces réseau qui peuvent être attachées à des instances Amazon EC2, vous pouvez utiliser la fonctionnalité de jonction d'interface réseau Elastic pour augmenter le nombre d'interfaces réseau disponibles. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois interfaces réseau. L'interface réseau principale de l'instance est considérée comme une interface réseau. Ainsi, vous pouvez attacher deux interfaces réseau supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une interface réseau, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Cela peut entraîner une sous-utilisation de la capacité de votre cluster. Si vous activez la jonction d'interface réseau Elastic, vous pouvez augmenter la densité de l'interface réseau afin de placer un plus grand nombre de tâches sur chaque instance. Lorsque la jonction est activée, une instance `c5.large` peut avoir jusqu'à 12 interfaces réseau. L'instance de conteneur a alors une interface réseau principale et Amazon ECS crée et attache une interface réseau « de jonction » à l'instance de conteneur. Par conséquent, avec cette configuration, vous pouvez exécuter 10 tâches sur l'instance au lieu des 2 tâches par défaut. Pour de plus amples informations, consultez la section Jonction d'interface réseau Elastic.

Limitation de l'API d'interface réseau Elastic

Lorsque vous exécutez des tâches qui utilisent le mode réseau `awsvpc`, Amazon ECS s'appuie sur les API Amazon EC2 suivantes. Chacune de ces API possède des limites d'API différentes. Pour plus d'informations, consultez la section Limitation des requêtes pour l'API Amazon EC2.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets

- DescribeSecurityGroups
- DescribeInstances

Si les appels d'API Amazon EC2 sont limités pendant les flux de travail d'allocation d'interface réseau Elastic, le planificateur de service Amazon ECS réessaie automatiquement avec des interruptions exponentielles. Ces tentatives automatiques peuvent parfois entraîner un retard dans le lancement des tâches, ce qui ralentit la vitesse de déploiement. En cas de limitation de l'API, le message `Operations are being throttled. Will try again later.` apparaît sur vos messages d'événement de service. Si vous respectez régulièrement les limites d'API Amazon EC2, vous pouvez nous contacter Support pour obtenir des conseils sur la manière d'augmenter vos limites de limitation d'API. Pour plus d'informations sur la surveillance et la résolution de ces erreurs de limitation, consultez la section Gestion des problèmes de limitation.

AWS Cloud Map

Amazon ECS Service Discovery utilise des AWS Cloud Map API pour gérer les espaces de noms de vos services Amazon ECS. Si vos services comportent un grand nombre de tâches, tenez compte des recommandations suivantes. Pour de plus amples informations, consultez la section Considérations en matière de découverte de service Amazon ECS.

AWS Cloud Map service quotas

Lorsque les services Amazon ECS sont configurés pour utiliser la découverte de services, le `Tasks per service` quota, qui correspond au nombre maximum de tâches pour le service, est affecté par le quota de AWS Cloud Map `Instances per service` service, qui est le nombre maximum d'instances pour ce service. En particulier, le quota de AWS Cloud Map service réduit le nombre de tâches que vous pouvez exécuter à un maximum de 1 000 tâches par service. Vous ne pouvez pas modifier le quota AWS Cloud Map . Pour plus d'informations, consultez Quotas de service AWS Cloud Map.

AWS Cloud Map Limitation d'API

Amazon ECS appelle les `DeregisterInstance` AWS Cloud Map API `ListInstances` `GetInstancesHealthStatusRegisterInstance`,, et en votre nom. Elles aident à la découverte de services et effectuent des surveillances de l'état lorsque vous lancez une tâche. Lorsque plusieurs services utilisant la découverte de services avec un grand nombre de tâches sont déployés en même temps, cela peut entraîner un dépassement des limites de limitation de l' AWS Cloud Map API. Lorsque cela se produit, vous verrez probablement le message suivant : `Operations are being`

`throttled. Will try again later` dans les messages d'événement de votre service Amazon ECS et vous constaterez un ralentissement du déploiement et de la vitesse de lancement des tâches. AWS Cloud Map ne documente pas les seuils de limitation pour ces API. Si vous rencontrez des problèmes de limitation à cause de ceux-ci, vous pouvez nous contacter Support pour obtenir des conseils sur l'augmentation des limites de limitation de votre API. Pour plus de recommandations sur la surveillance et le dépannage de telles erreurs de limitation, consultez la section Gestion des problèmes de limitation.

Gestion des problèmes de limitation d'Amazon ECS

Les erreurs de limitation sont classées en deux grandes catégories : la limitation synchrone et la limitation asynchrone.

Limitation synchrone

En cas de limitation synchrone, vous recevez immédiatement une réponse d'erreur d'Amazon ECS. Cette catégorie apparaît généralement lorsque vous appelez les API Amazon ECS lors de l'exécution de tâches ou de la création de services. Pour de plus amples informations sur la limitation impliquée et les limites d'accélération pertinentes, consultez la section Limitation de requête pour l'API Amazon ECS.

Lorsque votre application lance des demandes d'API, par exemple à l'aide du AWS CLI ou d'un AWS SDK, vous pouvez remédier à la limitation des API. Pour ce faire, vous pouvez soit concevoir l'architecture de votre application de manière à gérer les erreurs, soit implémenter une stratégie de backoff exponentiel et d'instabilité avec une logique de nouvelle tentative pour les appels d'API. Pour de plus amples informations, consultez la section Délais d'attente, nouvelles tentatives et backoff avec instabilité.

Si vous utilisez un AWS SDK, la logique de nouvelle tentative automatique est intégrée et configurable.

Limitation asynchrone

La limitation asynchrone se produit en raison de flux de travail asynchrones dans lesquels Amazon ECS ou CloudFormation pourrait appeler des API en votre nom pour provisionner des ressources. Il est important de savoir quelles AWS API Amazon ECS invoque en votre nom. Par exemple, l'API `CreateNetworkInterface` est invoquée pour les tâches qui utilisent le mode réseau `awsvpc`, et l'API `DescribeTargetHealth` est invoquée lors de l'exécution des surveillances de l'état pour les tâches enregistrées sur un équilibreur de charge.

Lorsque vos charges de travail atteignent une taille considérable, ces opérations d'API peuvent être limitées. En d'autres termes, ils sont peut-être suffisamment limités pour dépasser les limites imposées par Amazon ECS ou Service AWS ce que l'on appelle Amazon. Par exemple, si vous déployez des centaines de services, chacun comportant simultanément des centaines de tâches qui utilisent le mode réseau `awsvpc`, Amazon ECS invoque des opérations API Amazon EC2 telles que `CreateNetworkInterface` et des opérations API Elastic Load Balancing telles que `RegisterTarget` ou `DescribeTargetHealth` pour enregistrer respectivement l'interface réseau élastique et l'équilibreur de charge. Ces appels d'API peuvent dépasser les limites de l'API, ce qui entraîne des erreurs de limitation. Voici un exemple d'erreur de limitation d'Elastic Load Balancing incluse dans le message d'événement de service.

```
{
   "userIdentity":{
      "arn":"arn:aws:sts::111122223333:assumed-role/AWSServiceRoleForECS/ecs-service-
scheduler",
      "eventTime":"2022-03-21T08:11:24Z",
      "eventSource":"elasticloadbalancing.amazonaws.com",
      "eventName":" DescribeTargetHealth ",
      "awsRegion":"us-east-1",
      "sourceIPAddress":"ecs.amazonaws.com",
      "userAgent":"ecs.amazonaws.com",
      "errorCode":"ThrottlingException",
      "errorMessage":"Rate exceeded",
      "eventID":"0aeb38fc-229b-4912-8b0d-2e8315193e9c"
   }
}
```

Lorsque ces appels d'API partagent des limites avec le trafic d'autres API de votre compte, il peut être difficile de les surveiller, même s'ils sont émis sous forme d'événements de service.

Surveillance de la limitation

Il est important d'identifier quelles requêtes d'API sont limitées et qui les émet. Vous pouvez utiliser AWS CloudTrail ce qui surveille la limitation et s'intègre à CloudWatch Amazon Athena et Amazon. EventBridge Vous pouvez configurer CloudTrail pour envoyer des événements spécifiques aux CloudWatch journaux. CloudWatch Log Insights, analyse et analyse les événements. Cela permet d'identifier les détails des événements de limitation, tels que l'utilisateur ou le rôle IAM qui a effectué l'appel et le nombre d'appels d'API effectués. Pour plus d'informations, consultez la section Surveillance des fichiers CloudTrail journaux à l'aide de CloudWatch journaux.

Pour plus d'informations sur CloudWatch Logs Insights et des instructions sur la façon d'interroger les fichiers journaux, voir Analyse des données des CloudWatch journaux avec Logs Insights.

Amazon Athena vous permet de créer des requêtes et d'analyser des données à l'aide du langage SQL standard. Par exemple, vous pouvez créer une table Athena pour analyser les événements. CloudTrail Pour plus d'informations, consultez la section Utilisation de la CloudTrail console pour créer une table Athena pour les CloudTrail journaux.

Après avoir créé une table Athena, vous pouvez utiliser des requêtes SQL telles que la suivante pour examiner les erreurs `ThrottlingException`.

Remplacez le *`user-input`* par vos valeurs.

```
select eventname, errorcode,eventsource,awsregion, useragent,COUNT(*) count
FROM cloudtrail_table-name
where errorcode = 'ThrottlingException'
AND eventtime between '2024-09-24T00:00:08Z' and '2024-09-23T23:15:08Z'
group by errorcode, awsregion, eventsource, useragent, eventname
order by count desc;
```

Amazon ECS envoie également des notifications d'événements à Amazon EventBridge. Il existe des événements de modification de l'état des ressources et des événements d'action de service. Ils incluent des événements de limitation des API tels que `ECS_OPERATION_THROTTLED` et `SERVICE_DISCOVERY_OPERATION_THROTTLED`. Pour de plus amples informations, veuillez consulter Événements d'action d'un service Amazon ECS.

Ces événements peuvent être consommés par un service, par exemple AWS Lambda pour effectuer des actions en réponse. Pour de plus amples informations, veuillez consulter Gestion des événements Amazon ECS.

Si vous exécutez des tâches autonomes, certaines opérations API telles que `RunTask` sont asynchrones et les opérations de répétition ne sont pas automatiquement effectuées. Dans de tels cas, vous pouvez utiliser des services tels que EventBridge l'intégration pour réessayer des opérations limitées ou AWS Step Functions ayant échoué. Pour plus d'informations, consultez la section Gestion d'une tâche de conteneur (Amazon ECS, Amazon SNS).

CloudWatch À utiliser pour surveiller l'étranglement

CloudWatch propose une surveillance de l'utilisation de l'API sur l'`Usage`espace de noms sous Par AWS ressource. Ces métriques sont enregistrées avec le type API et le nom de la métrique

CallCount. Vous pouvez créer des alarmes qui démarreront chaque fois que ces mesures atteignent un certain seuil. Pour de plus amples informations, consultez la section Visualisation de vos quotas de service et définition des alarmes.

CloudWatch propose également la détection des anomalies. Cette fonctionnalité utilise le machine learning pour analyser et établir des bases de référence en fonction du comportement particulier de la métrique sur laquelle vous l'avez activée. En cas d'activité inhabituelle de l'API, vous pouvez utiliser cette fonctionnalité avec des CloudWatch alarmes. Pour plus d'informations, consultez la section Utilisation de la détection des CloudWatch anomalies.

En surveillant de manière proactive les erreurs de limitation, vous pouvez nous contacter Support pour augmenter les limites de limitation pertinentes et recevoir des conseils pour répondre aux besoins spécifiques de votre application.

Pratiques exemplaires en matière d'autoscaling et de gestion des capacités Amazon ECS

Vous pouvez exécuter des charges de travail d'applications conteneurisées de toutes tailles sur Amazon ECS. Cela inclut des environnements de test minimaux et des environnements de production de grande envergure qui fonctionnent à l'échelle mondiale.

Avec Amazon ECS, comme tous les AWS services, vous ne payez que ce que vous utilisez. Lorsque vous élaborez votre application de manière appropriée, vous pouvez réduire les coûts en ne consommant que les ressources dont vous avez besoin au moment où vous en avez besoin.

Les recommandations suivantes vous montrent comment exécuter vos charges de travail Amazon ECS afin d'atteindre vos objectifs de niveau de service tout en fonctionnant de manière rentable.

Rubriques

- Détermination de la taille d'une tâche pour Amazon ECS
- Optimisation de l'autoscaling d'un service Amazon ECS
- Capacité et disponibilité d'Amazon ECS
- Capacité de cluster Amazon ECS
- Choix de la taille des tâches Fargate pour Amazon ECS
- Accélération de l'allocation des capacités des clusters Amazon ECS avec les fournisseurs de capacité sur Amazon EC2

Détermination de la taille d’une tâche pour Amazon ECS

L’un des choix les plus importants lorsque vous déployez des conteneurs sur Amazon ECS concerne la taille de vos conteneurs et de vos tâches. La taille de vos conteneurs et de vos tâches est essentielle pour la mise à l’échelle et la planification des capacités.

Amazon ECS utilise deux métriques de ressources pour la capacité : l’UC et la mémoire. Amazon ECS mesure le processeur en unités 1/1024 d'un processeur virtuel complet (où 1 024 unités équivalent à 1 processeur virtuel complet). Amazon ECS mesure la mémoire en mégaoctets.

Dans la définition de votre tâche, vous pouvez déclarer des réserves et des limites de ressources.

Lorsque vous déclarez une réserve, vous déclarez la quantité minimale de ressources requise par une tâche. Votre tâche reçoit au moins la quantité de ressources que vous demandez. Votre application peut utiliser plus d’UC ou de mémoire que la réserve que vous déclarez. Cependant, cette capacité est soumise aux limites que vous avez également déclarées.

L’utilisation d’une quantité supérieure à la réserve est appelée éclatement (bursting). L’éclatement signifie que votre application utilise plus de ressources que ce que vous avez réservé, mais reste dans les limites que vous avez déclarées. Amazon ECS garantit les réserves. Par exemple, si vous utilisez des instances Amazon EC2 pour fournir de la capacité, Amazon ECS ne place aucune tâche sur une instance dans laquelle il ne peut pas honorer la réserve.

Une limite est la quantité maximale d’unités d’UC ou de mémoire que votre conteneur ou votre tâche peut utiliser. Si votre conteneur essaie d’utiliser plus d’UC que cette limite, Amazon ECS le limite. Si votre conteneur essaie d’utiliser plus de mémoire que cette limite, Amazon ECS l’arrête.

Choisir ces valeurs peut s’avérer difficile. Les valeurs les mieux adaptées à votre application dépendent dans une large mesure des besoins en ressources de votre application.

Le test de charge de votre application est la clé d’une planification réussie des besoins en ressources. Les tests de charge vous aident à mieux comprendre les exigences de votre application.

Applications sans état

Pour les applications sans état qui sont mises à l’échelle horizontalement, telles qu’une application située derrière un équilibreur de charge, nous vous recommandons de déterminer d’abord la quantité de mémoire consommée par votre application lorsqu’elle traite des requêtes.

Pour ce faire, vous pouvez utiliser des outils traditionnels tels que `ps` ou `top`. Vous pouvez également utiliser des solutions de surveillance telles que CloudWatch Container Insights.

Lorsque vous déterminez une réserve d'UC, réfléchissez à la manière dont vous souhaitez mettre votre application à l'échelle en fonction de vos besoins métier.

Vous pouvez utiliser des réservations de processeurs plus petites, telles que 256 unités de processeur (ou 1/4 vCPU), pour effectuer une mise à l'échelle de manière précise et minimiser les coûts. Mais ils risquent de ne pas se mettre à l'échelle assez rapidement pour répondre à des pics de demande importants.

Vous pouvez utiliser des réserves d'UC plus importantes pour procéder à une augmentation ou à une réduction horizontale plus rapidement. Cela vous permet de faire face plus rapidement aux pics de demande. Cependant, les réserves d'UCs plus importantes coûtent plus cher.

Autres applications

Pour les applications qui ne sont pas mises à l'échelle horizontalement, telles que les travailleurs individuels ou les serveurs de base de données, la capacité disponible et le coût sont vos principaux facteurs à prendre en compte.

Choisissez la quantité de mémoire et d'UC en fonction des tests de charge qui indiquent que vous avez besoin pour gérer le trafic et atteindre votre objectif de niveau de service. Amazon ECS garantit que votre application est placée sur un hôte doté d'une capacité adéquate.

Optimisation de l'autoscaling d'un service Amazon ECS

Un service Amazon ECS est un ensemble géré de tâches. Chaque service est associé à une définition de tâche, à un nombre de tâches souhaité et à une stratégie de placement facultative.

L'autoscaling d'un service Amazon ECS fonctionne via le service Application Auto Scaling. Application Auto Scaling utilise CloudWatch des métriques comme source pour les métriques de dimensionnement. Il utilise également des CloudWatch alarmes pour définir des seuils permettant d'augmenter ou de désactiver votre service.

Vous indiquez les seuils de mise à l'échelle. Vous pouvez définir une cible métrique, appelée mise à l'échelle du suivi de cible. Vous pouvez également spécifier des seuils, appelés mise à l'échelle par étapes.

Une fois que vous avez configuré Application Auto Scaling, celui-ci calcule en permanence le nombre de tâches souhaité pour le service. Il indique également à Amazon ECS lorsque le nombre de tâches souhaité doit changer, soit en l'augmentant, soit en le réduisant horizontalement.

Pour utiliser efficacement l'autoscaling du service, vous devez choisir une métrique de mise à l'échelle appropriée. Dans les sections suivantes, nous expliquons comment choisir une métrique.

Caractérisation de votre application

Pour dimensionner correctement une application, vous devez connaître les conditions dans lesquelles vous devez soit augmenter, soit réduire horizontalement votre application.

En substance, vous devez augmenter horizontalement votre application si la demande prévue dépasse la capacité disponible. À l'inverse, vous pouvez réduire horizontalement votre application pour réduire les coûts lorsque les ressources dépassent la demande.

Identification d'une métrique d'utilisation

Pour effectuer une mise à l'échelle efficace, vous devez identifier une métrique indiquant l'utilisation ou la saturation. Cette métrique doit présenter les propriétés suivantes pour être utile à la mise à l'échelle.

- La métrique doit être corrélée à la demande. Lorsque vous maintenez les ressources stables mais que la demande change, la valeur de la métrique doit également changer. La métrique doit augmenter ou diminuer lorsque la demande augmente ou diminue.
- La valeur de la métrique doit évoluer proportionnellement à la capacité. Lorsque la demande reste constante, l'ajout de ressources supplémentaires doit entraîner une modification proportionnelle de la valeur de la métrique. Ainsi, le doublement du nombre de tâches devrait entraîner une diminution de 50 % de la métrique.

Le meilleur moyen d'identifier une métrique d'utilisation consiste à effectuer des tests de charge dans un environnement de préproduction tel qu'un environnement intermédiaire. Les solutions de test de charge commerciales et open source sont largement disponibles. Ces solutions peuvent généralement générer une charge synthétique ou simuler le trafic utilisateur réel.

Pour démarrer le processus de test de charge, vous devez d'abord créer des tableaux de bord pour les indicateurs d'utilisation de votre application. Ces mesures incluent l'utilisation du processeur, l'utilisation de la mémoire, I/O les opérations, la profondeur de la I/O file d'attente et le débit réseau. Vous pouvez collecter ces statistiques à l'aide d'un service tel que CloudWatch Container Insights. Vous pouvez également les collecter en utilisant Amazon Managed Service for Prometheus avec Amazon Managed Grafana. Au cours de ce processus, veillez à collecter et à représenter graphiquement les métriques relatives aux temps de réponse ou aux taux d'achèvement des tâches de votre application.

Lorsque vous effectuez un test de charge, commencez par un faible taux de requêtes ou d'insertion de tâches. Maintenez ce débit pendant plusieurs minutes afin de permettre à votre application de prendre la température. Ensuite, augmentez lentement le taux et maintenez-le stable pendant quelques minutes. Répétez ce cycle en augmentant le taux à chaque fois jusqu'à ce que les délais de réponse ou de traitement de votre application soient trop lents pour atteindre vos objectifs de niveau de service (SLO).

Pendant le test de charge, examinez chacune des métriques d'utilisation. Les indicateurs qui augmentent en fonction de la charge sont les meilleurs candidats pour vous servir de meilleurs indicateurs d'utilisation.

Identifiez ensuite la ressource qui atteint la saturation. Dans le même temps, examinez également les métriques d'utilisation afin de déterminer laquelle se stabilise en premier à un niveau élevé. Vous pouvez également examiner laquelle atteint le pic, puis bloque votre application en premier. Par exemple, si l'utilisation de l'UC augmente de 0 % à 70-80 % à mesure que vous ajoutez de la charge, puis qu'elle reste à ce niveau une fois que vous ajoutez encore plus de charge, on peut affirmer sans risque de se tromper que l'UC est saturé. Selon l'architecture de l'UC, il se peut qu'il n'atteigne jamais 100 %. Supposons, par exemple, que l'utilisation de la mémoire augmente à mesure que vous ajoutez de la charge, puis que votre application se bloque soudainement lorsqu'elle atteint la limite de mémoire de la tâche ou de l'instance Amazon EC2. Dans ce cas, il est probable que la mémoire ait été entièrement consommée. Plusieurs ressources peuvent être consommées par votre application. Par conséquent, choisissez la métrique qui représente la ressource qui s'épuise en premier.

Enfin, essayez à nouveau le test de charge après avoir doublé le nombre de tâches ou d'instances Amazon EC2. Supposons que l'indicateur clé augmente ou diminue de moitié par rapport au taux précédent. Si tel est le cas, la métrique est proportionnelle à la capacité. Il s'agit d'un bon indicateur d'utilisation pour l'autoscaling.

Examinons maintenant ce scénario hypothétique. Supposons que vous soumettiez une application à un test de charge et que vous constatiez le résultat suivant : l'utilisation de l'UC atteint finalement 80 % à 100 requêtes par seconde. Lorsque vous ajoutez de la charge, l'utilisation de l'UC n'augmente plus. Cependant, cela ralentit la réponse de votre application. Ensuite, vous exécutez à nouveau le test de charge, en doublant le nombre de tâches tout en maintenant le taux à sa valeur maximale précédente. Si vous constatez que l'utilisation moyenne de l'UC tombe à environ 40 %, l'utilisation moyenne de l'UC est un bon candidat pour une métrique de mise à l'échelle. D'un autre côté, si l'utilisation de l'UC reste à 80 % après l'augmentation du nombre de tâches, l'utilisation moyenne de l'UC n'est pas une bonne métrique de mise à l'échelle. Dans ce cas, vous devrez effectuer des recherches supplémentaires afin de trouver une métrique appropriée.

Modèles d’application et propriétés de mise à l’échelle courants

Vous pouvez exécuter des logiciels de toutes sortes sur AWS. De nombreuses charges de travail sont développées en interne, tandis que d’autres sont basées sur des logiciels open source populaires. Quelle que soit leur origine, nous avons observé certains modèles de conception courants pour les services. La manière dont vous effectuez une mise à l’échelle efficace dépend en grande partie du modèle.

Le CPU-bound serveur efficace

Le CPU-bound serveur efficace n'utilise pratiquement aucune ressource autre que le processeur et le débit réseau. Chaque requête peut être traitée par l’application seule. Les requêtes ne dépendent pas d’autres services tels que les bases de données. L’application peut gérer des centaines de milliers de requêtes simultanées et peut utiliser efficacement plusieurs UC pour ce faire. Chaque requête est traitée soit par un thread dédié avec une faible surcharge de mémoire, soit par une boucle d’événements asynchrone qui s’exécute sur chaque UC qui traite les requêtes. Chaque réplica de l’application est également capable de traiter une requête. La seule ressource susceptible d’être épuisée avant l’UC est la bande passante du réseau. Dans les services liés à l’UC, l’utilisation de la mémoire, même au débit maximal, ne représente qu’une fraction des ressources disponibles.

Vous pouvez utiliser la mise à l'échelle CPU-based automatique pour ce type d'application. L’application bénéficie d’une flexibilité maximale en termes de mise à l’échelle. Vous pouvez le mettre à l’échelle verticalement en lui fournissant des instances Amazon EC2 plus grandes ou des vCPU Fargate. Et vous pouvez également la mettre à l’échelle horizontalement en ajoutant d’autres réplicas. L’ajout de réplicas supplémentaires, ou le doublement de la taille de l’instance, réduit de moitié l’utilisation moyenne de l’UC par rapport à la capacité.

Si vous utilisez la capacité Amazon EC2 pour cette application, pensez à la placer sur des instances optimisées pour le calcul, telles que la famille d’instances `c5` ou `c6g`

Serveur efficace lié à la mémoire

Le serveur efficace lié à la mémoire alloue une quantité importante de mémoire par requête. En cas de simultanéité maximale, mais pas nécessairement de débit, la mémoire est épuisée avant que les ressources d’UC ne soient épuisées. La mémoire associée à une requête est libérée lorsque celle-ci prend fin. Des requêtes supplémentaires peuvent être acceptées dans la limite de la mémoire disponible.

Vous pouvez utiliser l’autoscaling basé sur la mémoire pour ce type d’application. L’application bénéficie d’une flexibilité maximale en termes de mise à l’échelle. Vous pouvez la mettre à

l'échelle verticalement en lui fournissant des ressources de mémoire Amazon EC2 ou Fargate plus importantes. Et vous pouvez également la mettre à l'échelle horizontalement en ajoutant d'autres réplicas. L'ajout de réplicas supplémentaires ou le doublement de la taille de l'instance peut réduire de moitié l'utilisation moyenne de la mémoire par rapport à la capacité.

Si vous utilisez la capacité Amazon EC2 pour cette application, pensez à la placer sur des instances optimisées pour la mémoire, telles que la famille d'instances `r5` ou `r6g`.

Certaines applications limitées en mémoire ne libèrent pas la mémoire associée à une requête lorsqu'elle se termine, de sorte qu'une réduction de la simultanéité n'entraîne pas une réduction de la mémoire utilisée. Pour cela, nous vous déconseillons d'utiliser la mise à l'échelle basée sur la mémoire.

Le serveur basé sur les travailleurs

Le serveur basé sur les travailleurs traite une requête pour chaque thread de travail individuel l'une après l'autre. Les threads de travail peuvent être des threads légers, tels que des threads POSIX. Il peut également s'agir de threads plus lourds, tels que des processus UNIX. Quel que soit le type de thread, il existe toujours un nombre maximal de connexions simultanées que l'application peut prendre en charge. Généralement, la limite de simultanéité est définie proportionnellement aux ressources de mémoire disponibles. Si la limite de simultanéité est atteinte, l'application place des requêtes supplémentaires dans une file d'attente du backlog. Si la file d'attente du backlog est débordée, l'application rejette immédiatement les requêtes entrantes supplémentaires. Les applications courantes qui correspondent à ce modèle incluent le serveur Web Apache et Gunicorn.

La simultanéité des requêtes est généralement le meilleur indicateur pour dimensionner cette application. Étant donné qu'il existe une limite de simultanéité pour chaque réplica, il est important de procéder à une augmentation horizontale avant que la limite moyenne ne soit atteinte.

Le meilleur moyen d'obtenir des statistiques de simultanéité des demandes est de demander à votre application de les signaler à CloudWatch. Chaque réplica de votre application peut publier le nombre de requêtes simultanées sous forme de métrique personnalisée à une fréquence élevée. Nous recommandons de régler la fréquence au moins une fois par minute. Une fois que plusieurs rapports ont été collectés, vous pouvez utiliser la simultanéité moyenne comme métrique de mise à l'échelle. Vous calculez cette métrique en divisant la simultanéité totale par le nombre de réplicas. Par exemple, si la simultanéité totale est de 1 000 et que le nombre de réplicas est de 10, la simultanéité moyenne est de 100.

Si votre application se trouve derrière un Application Load Balancer, vous pouvez également utiliser la métrique `ActiveConnectionCount` de l'équilibreur de charge comme facteur dans la métrique

de mise à l'échelle. Vous devez diviser la métrique `ActiveConnectionCount` par le nombre de réplicas pour obtenir une valeur moyenne. Vous devez utiliser la valeur moyenne pour la mise à l'échelle, par opposition à la valeur de comptage brute.

Pour que cette conception fonctionne de manière optimale, l'écart type de la latence de réponse doit être faible à des taux de requêtes bas. Nous recommandons que, pendant les périodes de faible demande, la plupart des requêtes soient traitées dans un délai court et qu'il n'y ait pas beaucoup de requêtes dont le traitement prend beaucoup plus de temps que la moyenne. Le temps de réponse moyen devrait être proche du 95e centile du temps de réponse. Dans le cas contraire, des dépassements de files d'attente pourraient en résulter. Cela entraîne des erreurs. Nous vous recommandons de fournir des réplicas supplémentaires si nécessaire pour atténuer le risque de débordement.

Serveur en attente

Le serveur en attente effectue un certain traitement pour chaque requête, mais son fonctionnement dépend fortement d'un ou de plusieurs services en aval. Les applications de conteneurs font souvent un usage intensif des services en aval tels que les bases de données et autres services d'API. La réponse de ces services peut prendre un certain temps, en particulier dans les scénarios de haute capacité ou de concurrence élevée. En effet, ces applications ont tendance à utiliser peu de ressources d'UC et à utiliser leur simultanéité maximale en termes de mémoire disponible.

Le service en attente convient soit au modèle de serveur limité à la mémoire, soit au modèle de serveur basé sur le travailleur, selon la conception de l'application. Si la simultanéité de l'application n'est limitée que par la mémoire, l'utilisation moyenne de la mémoire doit être utilisée comme métrique de mise à l'échelle. Si la simultanéité de l'application est basée sur une limite de travail, la simultanéité moyenne doit être utilisée comme métrique de mise à l'échelle.

Le Java-based serveur

Si votre Java-based serveur est CPU-bound et évolue proportionnellement aux ressources du processeur, il peut être adapté à un modèle de CPU-bound serveur efficace. Si tel est le cas, l'utilisation moyenne de l'UC peut être appropriée comme métrique de mise à l'échelle. Cependant, de nombreuses applications Java ne le sont pas CPU-bound, ce qui rend leur mise à l'échelle difficile.

Pour de meilleures performances, nous vous recommandons d'allouer autant de mémoire que possible au segment de machine virtuelle Java (JVM). Les versions récentes de la JVM, y compris la mise à jour 191 ou ultérieure de Java 8, définissent automatiquement la taille du tas aussi grande que possible pour tenir dans le conteneur. Cela signifie qu'en Java, l'utilisation de la mémoire est rarement proportionnelle à l'utilisation des applications. À mesure que le taux de requêtes et

la simultanéité augmentent, l'utilisation de la mémoire reste constante. C'est pourquoi nous ne recommandons pas de dimensionner Java-based les serveurs en fonction de l'utilisation de la mémoire. Au lieu de cela, nous recommandons généralement une mise à l'échelle en fonction de l'utilisation de l'UC.

Dans certains cas, les Java-based serveurs sont confrontés à un épuisement du tas avant d'épuiser le processeur. Si votre application risque de s'épuiser en cas de forte simultanéité, le nombre moyen de connexions constitue la meilleure métrique de mise à l'échelle. Si votre application est susceptible de s'épuiser en tas à haut débit, le taux de requêtes moyen est la meilleure métrique de mise à l'échelle.

Serveurs qui utilisent d'autres environnements d'exécution avec récupérateur de mémoire

De nombreuses applications serveur sont basées sur des environnements d'exécution qui effectuent le récupérateur de mémoire, tels que .NET et Ruby. Ces applications serveur peuvent correspondre à l'un des modèles décrits précédemment. Cependant, comme pour Java, nous ne recommandons pas de dimensionner ces applications en fonction de la mémoire, car l'utilisation moyenne de la mémoire observée n'est souvent pas corrélée au débit ou à la simultanéité.

Pour ces applications, nous vous recommandons une mise à l'échelle en fonction de l'utilisation de l'UC si l'application est liée à l'UC. Dans le cas contraire, nous vous recommandons d'effectuer une mise à l'échelle en fonction du débit moyen ou de la simultanéité moyenne, en fonction des résultats de vos tests de charge.

Processeurs de tâches

De nombreuses charges de travail impliquent un traitement de tâches asynchrone. Il s'agit notamment des applications qui ne reçoivent pas de requêtes en temps réel, mais qui s'abonnent à une file d'attente pour recevoir des tâches. Pour ces types d'applications, la métrique de mise à l'échelle appropriée est presque toujours la profondeur de la file d'attente. La croissance des files d'attente indique que le travail en attente dépasse la capacité de traitement, tandis qu'une file d'attente vide indique qu'il y a plus de capacité que de travail à effectuer.

AWS les services de messagerie, tels qu'Amazon SQS et Amazon Kinesis Data Streams, fournissent des CloudWatch mesures qui peuvent être utilisées pour la mise à l'échelle. Pour Amazon SQS, `ApproximateNumberOfMessagesVisible` c'est la meilleure métrique. Pour Kinesis Data Streams, pensez à utiliser la métrique `MillisBehindLatest` publiée par la bibliothèque client Kinesis (KCL). Cette métrique doit être moyennée pour tous les consommateurs avant d'être utilisée à des fins de mise à l'échelle.

Capacité et disponibilité d’Amazon ECS

La disponibilité des applications est essentielle pour garantir une expérience sans erreur et minimiser le temps de latence des applications. La disponibilité dépend de l'accessibilité des ressources et d'une capacité suffisante pour répondre à la demande. AWS fournit plusieurs mécanismes pour gérer la disponibilité. Pour les applications que vous hébergez sur Amazon ECS, celles-ci incluent l’autoscaling et les zones de disponibilité (AZ). L’autoscaling gère le nombre de tâches ou d’instances en fonction des métriques que vous définissez, tandis que les zones de disponibilité vous permettent d’héberger votre application dans des emplacements isolés, mais géographiquement proches.

Comme pour la taille des tâches, la capacité et la disponibilité présentent certains compromis que vous devez prendre en compte. Idéalement, la capacité devrait être parfaitement adaptée à la demande. Il y aurait toujours juste assez de capacité pour traiter les requêtes et traiter les tâches afin d’atteindre les objectifs de niveau de service (SLO), notamment une faible latence et un faible taux d’erreur. La capacité ne devrait jamais être trop élevée, ce qui entraînerait des coûts excessifs, ni trop faible, ce qui entraînerait des temps de latence et des taux d’erreur élevés.

L’autoscaling est un processus latent. Tout d'abord, vous CloudWatch devez recevoir des métriques en temps réel. Il CloudWatch faut ensuite les agréger pour les analyser, ce qui peut prendre plusieurs minutes en fonction de la granularité de la métrique. CloudWatch compare les mesures aux seuils d'alarme pour identifier une pénurie ou un excès de ressources. Pour éviter toute instabilité, vous devez configurer les alarmes de manière à ce que le seuil défini soit dépassé pendant quelques minutes avant que l’alarme ne se déclenche. Il faut également du temps pour configurer de nouvelles tâches et mettre fin à des tâches qui ne sont plus nécessaires.

En raison de ces retards potentiels dans le système, vous devez conserver une certaine marge de manœuvre en cas de surprovisionnement. Over-provisioning peut aider à faire face à des pics de demande à court terme. Cela permet également à votre application de répondre à des requêtes supplémentaires sans atteindre la saturation. À titre de pratique exemplaire, vous pouvez définir votre objectif de mise à l’échelle entre 60 et 80 % d’utilisation. Cela permet à votre application de mieux gérer les pics de demande alors que de la capacité supplémentaire est encore en cours d’allocation.

Nous vous recommandons également de surprovisionner afin de pouvoir répondre rapidement aux défaillances de la zone de disponibilité. AWS recommande que les charges de travail de production soient gérées à partir de plusieurs zones de disponibilité. En effet, en cas de défaillance d’une zone de disponibilité, les tâches exécutées dans les zones de disponibilité restantes peuvent toujours répondre à la demande. Si votre application s’exécute dans deux zones de disponibilité, vous devez doubler le nombre normal de tâches. Cela vous permet de fournir une capacité immédiate en cas

de panne potentielle. Si votre application s'exécute dans trois zones de disponibilité, nous vous recommandons d'exécuter 1,5 fois le nombre normal de tâches. C'est-à-dire, exécutez trois tâches pour deux qui sont nécessaires au service ordinaire.

Maximisation de la vitesse de mise à l'échelle

L'autoscaling est un processus réactif dont l'effet prend du temps. Cependant, il existe des moyens de minimiser le temps nécessaire pour augmenter horizontalement.

Minimisez la taille de l'image. Le téléchargement et la décompression des images volumineuses à partir d'un référentiel d'images prennent plus de temps. Par conséquent, réduire la taille des images réduit le temps nécessaire au démarrage d'un conteneur. Pour réduire la taille de l'image, vous pouvez suivre ces recommandations spécifiques :

- Si vous pouvez créer un binaire statique ou utiliser Golang, créez votre image `FROM scratch` et n'incluez que votre application binaire dans l'image résultante.
- Utilisez des images de base minimisées provenant de fournisseurs de distribution en amont, tels qu'Amazon Linux ou Ubuntu.
- N'incluez aucun artefact de construction dans votre image finale. L'utilisation de versions en plusieurs étapes peut y contribuer.
- Dans la mesure du possible, compactez les étapes `RUN`. Chaque étape `RUN` crée une nouvelle couche d'image, ce qui entraîne un aller-retour supplémentaire pour télécharger la couche. Une étape `RUN` comportant plusieurs commandes jointes par `&&` comporte moins de couches qu'une étape comportant plusieurs étapes `RUN`.
- Si vous souhaitez inclure des données, telles que des données d'inférence ML, dans votre image finale, incluez uniquement les données nécessaires pour démarrer et commencer à desservir le trafic. Si vous récupérez des données à la demande depuis Amazon S3 ou un autre système de stockage sans affecter le service, stockez plutôt vos données dans ces emplacements.

Gardez vos images au plus proche. Plus la latence du réseau est élevée, plus le téléchargement de l'image prend du temps. Hébergez vos images dans un référentiel situé dans la même AWS région que celle dans laquelle se trouve votre charge de travail. Amazon ECR est un référentiel d'images hautes performances disponible dans toutes les régions dans lesquelles Amazon ECS est disponible. Évitez d'utiliser Internet ou un lien VPN pour télécharger des images de conteneurs. L'hébergement de vos images dans la même région améliore la fiabilité globale. Il atténue le risque de problèmes de connectivité réseau et de disponibilité dans une autre région. Vous pouvez également implémenter la réplication entre régions Amazon ECR pour y parvenir.

Réduisez les seuils de surveillance de l'état de l'équilibreur de charge. Les équilibreurs de charge effectuent une surveillance de l'état avant d'envoyer du trafic à votre application. La configuration de la surveillance de l'état par défaut pour un groupe cible peut prendre 90 secondes ou plus. Pendant ce temps, l'équilibreur de charge vérifie l'état et reçoit les requêtes. La réduction de l'intervalle entre les surveillances de l'état et du nombre de seuils peut permettre à votre application d'accepter le trafic plus rapidement et de réduire la charge sur les autres tâches.

Tenez compte des performances de démarrage à froid. Certaines applications utilisent des environnements d'exécution tels que la compilation Java perform Just-In-Time (JIT). Le processus de compilation, au moins lorsqu'il démarre, peut montrer les performances de l'application. Une solution consiste à réécrire les parties critiques de votre charge de travail en termes de latence dans des langages qui n'affectent pas les performances de démarrage à froid.

Utilisez la mise à l'échelle par étapes, et non des politiques de mise à l'échelle du suivi de cible. Vous disposez de plusieurs options d'Application Auto Scaling pour les tâches Amazon ECS. Le suivi des cibles est le mode le plus simple à utiliser. Il vous suffit de définir une valeur cible pour une métrique, telle que l'utilisation moyenne du processeur. Ensuite, l'autoscaler gère automatiquement le nombre de tâches nécessaires pour atteindre cette valeur. Grâce à la mise à l'échelle par étapes, vous pouvez réagir plus rapidement aux variations de la demande, car vous définissez les seuils spécifiques pour vos métriques de mise à l'échelle et le nombre de tâches à ajouter ou à supprimer lorsque les seuils sont dépassés. Et surtout, vous pouvez réagir très rapidement aux variations de la demande en réduisant la durée pendant laquelle une alarme de seuil est franchie. Pour plus d'informations, consultez Scalabilité automatique de service dans le Guide du développeur Amazon Elastic Container Service.

Si vous utilisez des instances Amazon EC2 pour fournir de la capacité de cluster, tenez compte des recommandations suivantes :

Utilisez des instances Amazon EC2 de plus grande taille et des volumes Amazon EBS plus rapides. Vous pouvez améliorer les vitesses de téléchargement et de préparation des images en utilisant une instance Amazon EC2 plus grande et un volume Amazon EBS plus rapide. Au sein d'une famille d'instances Amazon EC2 donnée, le débit maximal du réseau et d'Amazon EBS augmente à mesure que la taille de l'instance augmente (par exemple, de `m5.xlarge` à `m5.2xlarge`). En outre, vous pouvez également personnaliser les volumes Amazon EBS pour augmenter leur débit et leurs IOPS. Par exemple, si vous utilisez des volumes `gp2`, utilisez des volumes plus importants offrant un débit de base supérieur. Si vous utilisez des volumes `gp3`, spécifiez le débit et les IOPS lorsque vous créez le volume.

Utilisez le mode réseau en pont pour les tâches exécutées sur les instances Amazon EC2. Les tâches qui utilisent le mode réseau `bridge` sur Amazon EC2 démarrent plus rapidement que celles qui utilisent le mode réseau `awsvpc`. Lorsque le mode réseau `awsvpc` est utilisé, Amazon ECS attache une interface réseau Elastic (ENI) à l'instance avant de lancer la tâche. Cela introduit une latence supplémentaire. Il existe cependant plusieurs compromis à faire lors de l'utilisation d'un réseau de ponts. Ces tâches ne disposent pas de leur propre groupe de sécurité, ce qui a des répercussions sur l'équilibrage de charge. Pour plus d'informations, consultez la section Suppression de votre équilibreur de charge dans le Guide de l'utilisateur Elastic Load Balancing.

Gestion des pics de demande

Certaines applications subissent des pics soudains et importants en termes de demande. Cela se produit pour diverses raisons : un événement d'actualité, une grosse vente, un événement médiatique ou tout autre événement qui devient viral et entraîne une augmentation rapide et significative du trafic en très peu de temps. Si cela n'est pas planifié, la demande peut rapidement dépasser les ressources disponibles.

La meilleure façon de gérer les pics de demande est de les anticiper et de planifier en conséquence. La mise à l'échelle automatique pouvant prendre du temps, nous vous recommandons de faire évoluer votre application avant que le pic de demande ne commence. Pour de meilleurs résultats, nous recommandons d'élaborer un plan d'affaires impliquant une étroite collaboration entre les équipes utilisant un calendrier partagé. L'équipe chargée de planifier l'événement doit travailler en étroite collaboration avec l'équipe chargée de l'application à l'avance. Cela donne à cette équipe suffisamment de temps d'établir une planification claire. Ils peuvent planifier la capacité nécessaire pour augmenter horizontalement avant l'événement et la réduire horizontalement après l'événement. Pour plus d'informations, voir Mise à l'échelle planifiée dans le Guide de l'utilisateur Application Auto Scaling..

Si vous disposez d'un plan de support Entreprise, veillez également à collaborer avec votre responsable de compte technique (TAM). Votre TAM peut vérifier vos quotas de service et s'assurer que tous les quotas nécessaires sont augmentés avant le début de l'événement. De cette façon, vous n'atteignez aucun quota de service par accident. Ils peuvent également vous aider en préchauffant des services, tels que les équilibreurs de charge, afin de garantir le bon déroulement de votre événement.

Il est plus difficile de gérer les pics de demande imprévus. Les pics imprévus, s'ils sont d'une amplitude suffisante, peuvent rapidement faire en sorte que la demande dépasse la capacité. Cela peut également dépasser la capacité de réaction de l'autoscaling. La meilleure façon de se préparer

à des pics imprévus est de surallouer les ressources. Vous devez disposer de suffisamment de ressources pour répondre à la demande maximale de trafic prévue à tout moment.

Le maintien d'une capacité maximale en prévision de pics de demande imprévus peut s'avérer coûteux. Pour atténuer l'impact sur les coûts, trouvez une métrique avancée ou un événement qui permet de prédire qu'un pic de demande important est imminent. Si la métrique ou l'événement fournit un préavis significatif de manière fiable, lancez le processus d'augmentation horizontale immédiatement lorsque l'événement se produit ou lorsque la métrique dépasse le seuil spécifique que vous avez défini.

Si votre application est sujette à des pics de demande soudains et imprévus, envisagez d'ajouter un mode haute performance à votre application qui sacrifie les fonctionnalités non critiques tout en préservant les fonctionnalités cruciales pour le client. Supposons, par exemple, que votre application puisse passer de la génération de réponses personnalisées coûteuses à la diffusion d'une page de réponse statique. Dans ce scénario, vous pouvez augmenter le débit de manière significative sans devoir dimensionner l'application.

Enfin, vous pouvez envisager de dissocier les services monolithiques afin de mieux faire face aux pics de demande. Si votre application est un service monolithique coûteux à exécuter et lent à adapter, vous pourriez être en mesure d'extraire ou de réécrire des éléments critiques en termes de performances et de les exécuter en tant que services distincts. Ces nouveaux services peuvent ensuite être étendus indépendamment des composants moins critiques. Le fait de disposer de la flexibilité nécessaire pour augmenter horizontalement les fonctionnalités critiques en termes de performances séparément des autres parties de votre application peut à la fois réduire le temps nécessaire pour augmenter la capacité et contribuer à réduire les coûts.

Capacité de cluster Amazon ECS

Vous pouvez fournir de la capacité à un cluster Amazon ECS de plusieurs manières. Par exemple, vous pouvez lancer des instances Amazon EC2 et les enregistrer auprès du cluster au démarrage à l'aide de l'agent de conteneur Amazon ECS. Cependant, cette méthode peut s'avérer difficile, car vous devez gérer vous-même la mise à l'échelle. Par conséquent, nous vous recommandons d'utiliser les fournisseurs de capacité Amazon ECS. Les fournisseurs de capacités gèrent la mise à l'échelle des ressources pour vous. Il existe trois types de fournisseurs de capacité : Amazon EC2, Fargate et Fargate Spot. Pour plus d'informations sur les fournisseurs de capacité Fargate, consultez la section Clusters Amazon ECS pour les charges de travail Fargate et, pour les charges de travail EC2, consultez la section Clusters Amazon ECS pour les charges de travail EC2.

Les fournisseurs de capacité Fargate et Fargate Spot prennent en charge le cycle de vie des tâches Fargate pour vous. Fargate fournit une capacité à la demande et Fargate Spot fournit une capacité Spot. Lorsque vous lancez une tâche, Amazon ECS met à votre disposition une ressource Fargate. Cette ressource Fargate est fournie avec les unités de mémoire et d'UC qui correspondent directement aux limites de niveau de tâche que vous avez déclarées dans votre définition de tâche. Chaque tâche reçoit sa propre ressource Fargate, ce qui crée une relation 1:1 entre la tâche et les ressources de calcul.

Les tâches exécutées sur Fargate Spot sont susceptibles d'être interrompues. Les interruptions surviennent après un avertissement de deux minutes. Elles se produisent pendant les périodes de forte demande. Fargate Spot convient parfaitement aux charges de travail tolérantes aux interruptions, telles que les tâches par lots, les environnements de développement ou de préparation. Ils sont également adaptés à tout autre scénario où une haute disponibilité et une faible latence ne sont pas requises.

Vous pouvez exécuter des tâches Fargate Spot parallèlement à des tâches Fargate à la demande. En les utilisant ensemble, vous bénéficiez d'une capacité « de débordement » pour l'allocation à moindre coût.

Amazon ECS peut également gérer la capacité de l'instance Amazon EC2 pour vos tâches. Chaque fournisseur de capacité Amazon EC2 est associé à un groupe Amazon EC2 Auto Scaling que vous spécifiez. Lorsque vous utilisez le fournisseur de capacité Amazon EC2, l'autoscaling de cluster maintient la taille du groupe Amazon EC2 Auto Scaling afin de garantir que toutes les tâches planifiées peuvent être placées.

Choix de la taille des tâches Fargate pour Amazon ECS

Pour les définitions de AWS Fargate tâches, vous devez spécifier le processeur et la mémoire au niveau de la tâche, et vous n'avez pas besoin de prendre en compte les frais généraux. Vous pouvez également spécifier l'UC et la mémoire au niveau du conteneur pour les tâches Fargate. Toutefois, cette opération n'est pas obligatoire. Les limites de ressources doivent être supérieures ou égales à toutes les réserves que vous avez déclarées. Dans la plupart des cas, vous pouvez les définir en fonction de la somme des réserves de chacun des conteneurs déclarés dans votre définition de tâche. Ensuite, arrondissez le nombre à la taille Fargate la plus proche. Pour plus d'informations sur les tailles disponibles, consultez la section UC et mémoire de la tâche..

Accélération de l'allocation des capacités des clusters Amazon ECS avec les fournisseurs de capacité sur Amazon EC2

Les clients qui exécutent Amazon ECS sur Amazon EC2 peuvent tirer parti de l'autoscaling de cluster (CAS, Cluster Auto Scaling) Amazon ECS pour gérer la mise à l'échelle des groupes Amazon EC2 Auto Scaling (ASG, Auto Scaling groups). Avec le CAS, vous pouvez configurer Amazon ECS pour mettre automatiquement votre ASG à l'échelle et vous concentrer uniquement sur l'exécution de vos tâches. Amazon ECS veillera à ce que l'ASG évolue en fonction des besoins, sans qu'aucune autre intervention ne soit requise. Les fournisseurs de capacité Amazon ECS vous permettent de gérer l'infrastructure de votre cluster en s'assurant qu'il y a suffisamment d'instances de conteneur pour répondre aux demandes de votre application. Pour en savoir plus sur le fonctionnement du CAS Amazon ECS, consultez le billet de blog Deep Dive on Amazon ECS Cluster Auto Scaling.

Étant donné que CAS repose sur une intégration CloudWatch basée avec ASG pour ajuster la capacité du cluster, il présente une latence inhérente associée à la publication CloudWatch des métriques, au temps nécessaire pour que la métrique franchisse les CloudWatch alarmes (`CapacityProviderReservation`à la fois élevé et faible) et au temps nécessaire au préchauffage d'une instance Amazon EC2 récemment lancée. Vous pouvez prendre les mesures suivantes pour améliorer la réactivité du CAS et accélérer les déploiements :

Mise à l'échelle par étapes du fournisseur de capacités

Les fournisseurs de capacité Amazon ECS finiront par grow/shrink créer des instances de conteneurs pour répondre aux exigences de votre application. Le nombre minimum d'instances qu'Amazon ECS lancera est fixé à une par défaut. Cela peut allonger la durée de vos déploiements si plusieurs instances sont nécessaires pour placer vos tâches en attente. Vous pouvez augmenter la `minimumScalingStepSize` en utilisant l'API Amazon ECS pour augmenter le nombre minimum d'instances qu'Amazon ECS fait évoluer ou non à la fois. Une `maximumScalingStepSize` trop faible peut limiter le nombre d'instances de conteneur pouvant être mises à l'échelle horizontale, ce qui peut ralentir vos déploiements.

> **Note**
>
> Cette configuration n'est actuellement disponible qu'en utilisant les API `CreateCapacityProvider` ou `UpdateCapacityProvider`.

Durée de préinitialisation de l’instance

La période de préchauffage de l'instance est la période au terme de laquelle une instance Amazon EC2 récemment lancée peut contribuer aux CloudWatch métriques du groupe Auto Scaling. Une fois la période de préchauffage spécifiée expirée, l’instance est prise en compte dans les métriques agrégées de l’ASG, et le CAS procède à sa prochaine itération de calculs pour estimer le nombre d’instances requises.

La valeur par défaut pour `instanceWarmupPeriod` est de 300 secondes, que vous pouvez configurer à une valeur inférieure en utilisant les API `CreateCapacityProvider` ou `UpdateCapacityProvider` pour une mise à l’échelle plus réactive.

Capacité de réserve

Si votre fournisseur de capacité ne dispose d’aucune instance de conteneur pour placer des tâches, il doit accroître (augmenter horizontalement) la capacité du cluster en lançant des instances Amazon EC2 à la volée, puis attendre leur démarrage avant de pouvoir y lancer des conteneurs. Cela peut réduire considérablement la vitesse de lancement des tâches. Vous avez deux options.

Dans ce cas, le fait de disposer d’une capacité Amazon EC2 disponible déjà lancée et prête à exécuter des tâches augmentera la vitesse de lancement effective des tâches. Vous pouvez utiliser la configuration de `Target Capacity` pour indiquer que vous souhaitez conserver de la capacité de réserve dans vos clusters. Par exemple, en définissant `Target Capacity` sur 80 %, vous indiquez que votre cluster a besoin de 20 % de capacité de réserve à tout moment. Cette capacité de réserve peut permettre le lancement immédiat de toutes les tâches autonomes, garantissant ainsi que les lancements de tâches ne sont pas ralentis. L’inconvénient de cette approche est l’augmentation potentielle des coûts liés au maintien de la capacité de réserve des clusters.

Une autre approche que vous pouvez envisager consiste à augmenter la marge de manœuvre de votre service, et non celle du fournisseur de capacité. Cela signifie qu’au lieu de réduire la configuration de `Target Capacity` pour lancer de la capacité de réserve, vous pouvez augmenter le nombre de réplicas dans votre service en modifiant la métrique de mise à l’échelle du suivi de cible ou les seuils de mise à l’échelle par étapes de l’autoscaling du service. Notez que cette approche ne sera utile que pour les charges de travail complexes, mais n’aura aucun effet lorsque vous déployez de nouveaux services et que vous passez de 0 à N tâches pour la première fois. Pour plus d’informations sur les politiques de mise à l’échelle associées, consultez les sections Politiques de mise à l’échelle du suivi de cible ou Politiques de mise à l’échelle par étapes.

Accès aux fonctionnalités d’Amazon ECS avec les paramètres du compte

Vous pouvez accéder aux paramètres de compte Amazon ECS pour activer ou désactiver des fonctions spécifiques. Pour chaque Région AWS, vous pouvez accepter ou refuser chaque paramètre de compte au niveau du compte ou pour un utilisateur ou un rôle spécifique.

Vous pouvez choisir d'activer ou de désactiver des fonctions spécifiques si l'une des options suivantes vous concerne :

- Un utilisateur ou un rôle peut accepter ou refuser des paramètres de compte spécifiques pour son compte individuel.
- Un utilisateur ou un rôle peut définir le paramètre d'acceptation et de refus par défaut pour tous les utilisateurs du compte.
- L’utilisateur racine ou un utilisateur disposant de privilèges d’administrateur peut choisir d’accepter ou de refuser un rôle ou un utilisateur spécifique sur le compte. Si le paramètre de compte de l'utilisateur root est modifié, cela modifie également le paramètre par défaut de tous les utilisateurs et rôles pour lesquels aucun paramètre de compte individuel n'a été sélectionné.

> **Note**
>
> Les utilisateurs fédérés héritent du paramètre de compte de l'utilisateur root et ne peuvent pas avoir de paramètres de compte définis séparément pour leur compte.

Les paramètres de compte suivants sont disponibles. Vous devez activer et désactiver séparément chaque paramètre de votre compte.

Nom de la ressource	En savoir plus
`containerInsights`	Container Insights

`serviceLongArnFormat`
`taskLongArnFormat`
`containerInstanceLongArnFormat`
	Amazon Resource Names (ARN) et ID

Nom de la ressource	En savoir plus
`tagResourceAuthorization`	Autorisation de balisage
`fargateFIPSMode`	AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)
`fargateTaskRetirementWaitPeriod`	AWS Fargate temps d'attente pour le retrait d'une tâche
`fargateEventWindows`	AWS Fargate retraits de tâches à l'aide des fenêtres d'événements EC2
`guardDutyActivate`	Surveillance de l'exécution (GuardDuty intégration Amazon)
`dualStackIPv6`	VPC IPv6 à double pile
`awsvpcTrunking`	Augmentation des interfaces réseau des instances de conteneur Linux
`defaultLogDriverMode`	Mode de pilote du journal par défaut

Amazon Resource Names (ARN) et ID

Lorsque des ressources Amazon ECS sont créées, chacune d'elles se voit affecter un Amazon Resource Name (ARN) et un identificateur de ressource (ID) uniques. Si vous utilisez un outil de ligne de commande ou l'API Amazon ECS pour gérer Amazon ECS, des ARN ou des ID de ressource sont requis pour certaines commandes. Par exemple, si vous utilisez la AWS CLI commande stop-task pour arrêter une tâche, vous devez spécifier l'ARN ou l'ID de la tâche dans la commande.

Amazon ECS a introduit un nouveau format pour les Amazon Resource Names (ARN) et les identifiants de ressources pour les services, tâches et instances de conteneurs Amazon ECS. L'état d'acceptation de chaque type de ressource détermine le format Amazon Resource Name (ARN) utilisé par la ressource. Vous devez accepter le nouveau format ARN afin d'utiliser des fonctions telles que le balisage des ressources pour ce type de ressource.

Vous pouvez accepter ou refuser le nouveau nom Amazon Resource Name (ARN) et les ID de ressource dépend de la région. Actuellement, il est activé par défaut pour tout nouveau compte.

Vous pouvez accepter ou refuser le nouveau format d'Amazon Resource Name (ARN) et d'ID de ressource à tout moment. Une fois que vous avez activé cette option, toutes les nouvelles ressources que vous créez utilisent le nouveau format.

> **Note**
>
> Un ID de ressource ne change pas une fois qu'il a été créé. Par conséquent, l'option d'entrée ou de retrait du nouveau format n'affecte pas vos ID de ressource existants.

Les sections suivantes décrivent les modifications des formats d'ARN et d'ID de ressource. Pour plus d'informations sur la transition vers les nouveaux formats, consultez la FAQ sur Amazon Elastic Container Service.

Format Amazon Resource Name (ARN)

Certaines ressources ont un nom convivial, comme par exemple un service nommé `production`. Dans d'autres cas, vous devez définir une ressource à l'aide du format Amazon Resource Name (ARN). Le nouveau format ARN des tâches, services et instances de conteneur Amazon ECS inclut le nom du cluster. Pour de plus amples informations sur le nouveau format ARN, consultez Modification des paramètres de compte Amazon ECS.

Le tableau suivant présente le format actuel et le nouveau format pour chaque type de ressource :

Type de ressource	ARN
Instance de conteneur	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`container-instance-id`* actuel Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:container-instance/` *`cluster-name`* `/`*`container-instance-id`*
Amazon ECS service	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`service-name`* actuel Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:service/` *`cluster-name`* `/`*`service-name`*

Type de ressource	ARN
Tâches Amazon ECS	`arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`task-id`* actuel Nouveau : `arn:aws:ecs:` *`region`*`:`*`aws_account_id`* `:task/`*`cluster-name`*`/`*`task-id`*

Longueur des ID de ressource

Un ID de ressource est constitué d'une combinaison unique de lettres et de chiffres. Les nouveaux formats d'ID de ressource incluent des ID plus courts pour les tâches et les instances de conteneur Amazon ECS. Le format d'ID de ressource actuel comporte 36 caractères. Le nouveau format d'ID comporte 32 caractères sans traits d'union. Pour de plus amples informations sur le nouveau format d'ID de ressource, consultez Modification des paramètres de compte Amazon ECS.

La valeur par défaut est `enabled`.

Seules les ressources lancées après l'acceptation recevront le nouveau format d'ARN et d'ID de ressource. Toutes les ressources existantes ne sont pas affectées. Pour que les services et les tâches Amazon ECS passent aux nouveaux formats d'ARN et d'ID de ressource, vous devez recréer les services et les tâches. Pour faire passer une instance de conteneur au nouveau format d'ARN et d'ID de ressource, il est nécessaire de la vider et de lancer et d'enregistrer une nouvelle instance de conteneur sur le cluster.

> **Note**
>
> Les tâches lancées par un service Amazon ECS ne peuvent recevoir le nouveau format d'ARN et d'ID de ressource que si le service a été créé à compter du 16 novembre 2018 et si l'utilisateur qui a créé le service a accepté le nouveau format pour les tâches.

Calendrier relatif au format ARN et de l'ID de ressource

Le calendrier prévoyant des créneaux d'activation/rejet du nouveau format d'Amazon Resource Name (ARN) et d'ID de ressource pour les ressources Amazon ECS a pris fin le 1er avril 2021. Le nouveau format est activé par défaut pour tout nouveau compte. Toutes les nouvelles ressources créées reçoivent le nouveau format et vous ne pouvez plus vous désinscrire.

Container Insights

Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge une observabilité améliorée pour les clusters Amazon ECS utilisant les instances gérées Amazon ECS Fargate et EC2. Une fois que vous avez configuré Container Insights avec observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques d'infrastructure détaillées depuis le niveau du cluster jusqu'au niveau du conteneur dans votre environnement et affiche vos données dans des tableaux de bord qui vous présentent diverses métriques et dimensions. Vous pouvez ensuite utiliser ces tableaux de bord prêts à l'emploi dans la console Container Insights pour mieux comprendre l'état et les performances de votre conteneur, et pour réduire les problèmes plus rapidement en identifiant les anomalies.

Nous vous recommandons d'utiliser Container Insights avec observabilité améliorée à la place de Container Insights, car il fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution. Pour plus d'informations, consultez la section Métriques Amazon ECS Container Insights avec observabilité améliorée dans le Guide de l'utilisateur Amazon CloudWatch .

La valeur par défaut du paramètre du compte `containerInsights` est `disabled`.

Container Insights avec observabilité améliorée

Utilisez la commande suivante pour activer Container Insights avec observabilité améliorée.

Réglez le paramètre du compte `containerInsights` sur `enhanced`.

```
aws ecs put-account-setting --name containerInsights --value enhanced
```

Exemple de sortie

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enhanced",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Une fois que vous avez défini ce paramètre de compte, tous les nouveaux clusters utilisent automatiquement Container Insights avec observabilité améliorée. Utilisez la commande `update-cluster-settings` pour ajouter Container Insights avec observabilité améliorée à un cluster existant, ou pour mettre à niveau un cluster de Container Insights à Container Insights avec observabilité améliorée.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enhanced
```

Vous pouvez également utiliser la console pour configurer Container Insights avec observabilité améliorée. Pour de plus amples informations, veuillez consulter Modification des paramètres de compte Amazon ECS.

Container Insights

Lorsque vous définissez le paramètre de compte `containerInsights` sur `enabled`, tous les nouveaux clusters ont Container Insights activé par défaut. Vous pouvez modifier les clusters existants en utilisant `update-cluster-settings`.

Pour utiliser Container Insights, définissez le paramètre du compte `containerInsights` sur `enabled`. Utilisez la commande suivante pour activer Container Insights.

```
aws ecs put-account-setting --name containerInsights --value enabled
```

Exemple de sortie

```
{
    "setting": {
        "name": "containerInsights",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:johndoe",
         "type": user
    }
}
```

Lorsque vous définissez le paramètre de compte `containerInsights` sur `enabled`, tous les nouveaux clusters ont Container Insights activé par défaut. Utilisez la commande `update-cluster-settings` pour ajouter Container Insights à un cluster existant.

```
aws ecs update-cluster-settings --cluster cluster-name --settings
 name=containerInsights,value=enabled
```

Vous pouvez également utiliser la console pour configurer Container Insights. Pour de plus amples informations, veuillez consulter Modification des paramètres de compte Amazon ECS.

AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)

Fargate prend en charge la norme fédérale de traitement de l'information (FIPS-140) qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent les informations sensibles. Il s'agit de la norme gouvernementale en vigueur aux États-Unis et au Canada, applicable aux systèmes qui doivent être conformes à la loi sur la gestion de la sécurité des informations fédérales (FISMA) ou au programme fédéral de gestion des risques et des autorisations (FedRAMP).

Le nom de la ressource est `fargateFIPSMode`.

La valeur par défaut est `disabled`.

Vous devez activer la conformité à la Federal Information Processing Standard (FIPS-140) sur Fargate. Pour de plus amples informations, veuillez consulter the section called “Conformité d'AWS Fargate FIPS-140 ”.

> **Important**
>
> Le paramètre de compte `fargateFIPSMode` ne peut être modifié qu'à l'aide de l'API Amazon ECS ou de la AWS CLI. Pour de plus amples informations, veuillez consulter Modification des paramètres de compte Amazon ECS.

Exécutez `put-account-setting-default` avec l'option `fargateFIPSMode` définie à `enabled`. Pour plus d'informations, consultez put-account-setting-default dans le Guide de référence des API Amazon Elastic Container Service.

- Vous pouvez utiliser la commande suivante pour activer la FIPS-140 conformité.

  ```
  aws ecs put-account-setting-default --name fargateFIPSMode --value enabled
  ```

  Exemple de sortie

```
{
    "setting": {
        "name": "fargateFIPSMode",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
         "type": user
    }
}
```

Vous pouvez exécuter `list-account-settings` pour afficher l'état de FIPS-140 conformité actuel. Utilisez l'option `effective-settings` pour afficher les paramètres au niveau du compte.

```
aws ecs list-account-settings --effective-settings
```

Autorisation de balisage

Amazon ECS introduit l'autorisation de balisage pour la création de ressources. Les utilisateurs doivent disposer des autorisations d'étiquetage pour les actions qui créent la ressource, telles que `ecsCreateCluster`. Lorsque vous créez une ressource et que vous spécifiez des balises pour cette ressource, AWS effectue une autorisation supplémentaire pour vérifier qu'il existe des autorisations pour créer des balises. Par conséquent, vous devez octroyer des autorisations explicites d'utiliser l'action `ecs:TagResource`. Pour de plus amples informations, veuillez consulter the section called "Baliser des ressources pendant la création".

Pour activer l'autorisation de balisage, exécutez `put-account-setting-default` avec l'option `tagResourceAuthorization` définie sur on. Pour plus d'informations, consultez put-account-setting-default dans le Guide de référence des API Amazon Elastic Container Service. Vous pouvez exécuter `list-account-settings` pour afficher l'état actuel de l'autorisation de balisage.

- Vous pouvez utiliser la commande suivante pour activer l'autorisation d'étiquetage.

```
aws ecs put-account-setting-default --name tagResourceAuthorization --value on --
region region
```

Exemple de sortie

```
{
    "setting": {
```

```
        "name": "tagResourceAuthorization",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": user
    }
}
```

Une fois l'adhésion effectuée, vous devez configurer les autorisations appropriées pour permettre aux utilisateurs d'étiqueter les ressources lors de leur création. Pour de plus amples informations, veuillez consulter the section called "Baliser des ressources pendant la création".

Vous pouvez exécuter `list-account-settings` pour afficher l'état actuel de l'autorisation de balisage. Utilisez l'option `effective-settings` pour afficher les paramètres au niveau du compte.

```
aws ecs list-account-settings --effective-settings
```

Chronologie des autorisations de balisage

Vous pouvez vérifier si l'autorisation de balisage est active en exécutant `list-account-settings` pour afficher la valeur `tagResourceAuthorization`. Lorsque la valeur est égale à on, cela signifie que l'autorisation de balisage est active. Pour plus d'informations, consultez list-account-setting dans le Guide de référence des API Amazon Elastic Container Service.

Voici les dates importantes concernant l'autorisation de balisage.

- 18 avril 2023 : introduction de l'autorisation de balisage. Tous les comptes nouveaux et existants doivent adhérer pour utiliser la fonctionnalité. Vous pouvez choisir d'activer l'autorisation d'étiquetage. En adhérant, vous devez octroyer les autorisations appropriées.
- 9 février 2024 – 6 mars 2024 : l'autorisation d'étiquetage est activée par défaut pour tous les nouveaux comptes et les comptes existants non concernés. Vous pouvez activer ou désactiver les paramètres du compte `tagResourceAuthorization` pour vérifier votre politique IAM.

  AWS a notifié les comptes concernés.

  Pour désactiver la fonctionnalité, exécutez `put-account-setting-default` avec l'option `tagResourceAuthorization` définie sur `off`.
- 7 mars 2024 : si vous avez activé l'autorisation d'étiquetage, vous ne pouvez plus désactiver les paramètres du compte.

Nous vous recommandons de terminer le test de votre politique IAM avant cette date.

- 29 mars 2024 : tous les comptes utilisent l'autorisation d'étiquetage. Le paramètre au niveau du compte ne sera plus disponible dans la console Amazon ECS ou l' AWS CLI.

AWS Fargate temps d'attente pour le retrait d'une tâche

AWS envoie des notifications lorsque des tâches Fargate sont en cours d'exécution sur une version de la plateforme dont la révision est destinée à être supprimée. Pour de plus amples informations, veuillez consulter Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.

AWS est responsable de la mise à jour et de la maintenance de l'infrastructure sous-jacente de AWS Fargate. Lorsqu'il est AWS déterminé qu'une mise à jour de sécurité ou d'infrastructure est nécessaire pour une tâche Amazon ECS hébergée sur Fargate, les tâches doivent être arrêtées et de nouvelles tâches doivent être lancées pour les remplacer. Vous pouvez configurer la période d'attente avant que les tâches ne soient retirées pour être corrigées. Vous avez la possibilité de mettre la tâche hors service immédiatement, d'attendre 7 jours calendaires ou d'attendre 14 jours calendaires.

Ce paramètre se trouve au niveau du compte.

Vous pouvez configurer l'heure à laquelle Fargate commence le retrait des tâches. Le délai d'attente par défaut est de 7 jours. Pour les charges de travail qui nécessitent l'application immédiate des mises à jour, choisissez le paramètre immédiat (`0`). Si vous avez besoin de plus de temps, configurez l'option`7`, ou `14` jour.

Nous vous recommandons de choisir une période d'attente plus courte afin de pouvoir accéder plus rapidement aux nouvelles versions de plateforme.

Configurez la période d'attente en exécutant `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur racine ou utilisateur administratif. Utilisez l'option `fargateTaskRetirementWaitPeriod` pour le `name` et l'option `value` définie sur l'une des valeurs suivantes :

- `0`- AWS envoie la notification et commence immédiatement à supprimer les tâches concernées.
- 7- AWS envoie la notification et attend 7 jours calendaires avant de commencer à retirer les tâches concernées. Il s'agit de l'option par défaut.
- `14` : AWS envoie la notification et attend 14 jours calendaires avant de commencer à retirer les tâches concernées.

Pour plus d'informations, veuillez consulter put-account-setting-default et put-account-setting dans la Référence de l'API Amazon Elastic Container Service.

Vous pouvez exécuter la commande suivante afin de définir le délai d'attente à 14 jours.

```
aws ecs put-account-setting-default --name fargateTaskRetirementWaitPeriod --value 14
```

Exemple de sortie

```
{
    "setting": {
        "name": "fargateTaskRetirementWaitPeriod",
        "value": "14",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type: user"
    }
}
```

Vous pouvez exécuter `list-account-settings` pour afficher le temps d'attente actuel pour retirer la tâche Fargate. Utilisez l'option `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

AWS Fargate retraits de tâches à l'aide des fenêtres d'événements EC2

AWS est responsable de l'application des correctifs et de la maintenance de l'infrastructure sous-jacente pour AWS Fargate. Quand AWS détermine qu'une mise à jour de sécurité ou d'infrastructure est nécessaire pour une tâche Amazon ECS hébergée sur Fargate, les tâches doivent être arrêtées et de nouvelles tâches lancées pour les remplacer. En activant ce paramètre de compte, AWS Fargate vous utiliserez les fenêtres d'événements EC2 pour déterminer l'heure et le jour auxquels vos tâches seront supprimées. Notez qu'il s'agit d'une activation unique pour un compte.

Pour activer l'utilisation des fenêtres d'événements EC2 pour les retraits de tâches Fargate, vous pouvez utiliser la commande CLI suivante :

```
aws ecs put-account-setting-default --name fargateEventWindows --value enabled
```

Il s'agit d'un paramètre au niveau du compte. Vous pouvez associer des fenêtres d'événements EC2 à des tâches Fargate au niveau du compte, du cluster et du service à l'aide des balises d'instance suivantes :

- `aws:ecs:serviceArn`: pour les tâches de service
- `aws:ecs:clusterArn`: pour les tâches appartenant à un cluster
- `aws:ecs:fargateTask`: réglé sur true pour cibler toutes les tâches Fargate du compte

Pour en savoir plus sur le fonctionnement des fenêtres d'événements Amazon EC2 pour vos tâches Amazon ECS exécutées sur Fargate, consultez Étape 1 : définissez le temps d'attente de la tâche ou utilisez les fenêtres d'événements Amazon EC2

Augmentation des interfaces réseau des instances de conteneur Linux

Les tâches Amazon ECS qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'interfaces réseau qui peuvent être attachées à des instances Amazon EC2 est limité par défaut et l'interface réseau principale est considérée comme l'une d'elles. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI qui lui sont attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez donc attacher deux ENI supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance.

Amazon ECS prend en charge le lancement d'instances de conteneur avec une augmentation de la densité ENI et utilisant des types d'instances Amazon EC2 pris en charge. Lorsque vous utilisez ces types d'instances et activez le paramètre de compte `awsvpcTrunking`, des ENI supplémentaires sont disponibles sur les instances de conteneur nouvellement lancées. Cette configuration vous permet de placer plus de tâches sur chaque instance de conteneur.

Par exemple, une instance `c5.large` dotée du paramètre `awsvpcTrunking` dispose d'une limite ENI augmentée à douze. L'instance de conteneur a alors une interface réseau principale. Amazon ECS crée et attache une interface réseau « tronc » à l'instance de conteneur. Par conséquent, cette configuration vous permet de lancer dix tâches sur l'instance de conteneur au lieu des deux tâches actuellement possibles.

Surveillance de l'exécution (GuardDuty intégration Amazon)

Runtime Monitoring est un service intelligent de détection des menaces qui protège les charges de travail exécutées sur des instances de conteneurs Fargate et EC2 en surveillant en permanence l'activité des AWS journaux et du réseau afin d'identifier les comportements malveillants ou non autorisés.

Le `guardDutyActivate` paramètre est en lecture seule dans Amazon ECS et indique si la surveillance de l'exécution est activée ou désactivée par votre administrateur de sécurité dans votre compte Amazon ECS. GuardDuty contrôle les paramètres de ce compte en votre nom. Pour plus d'informations, consultez la section Protection des charges de travail Amazon ECS à l'aide de la surveillance d'exécution.

Vous pouvez exécuter `list-account-settings` pour afficher le paramètre GuardDuty d'intégration actuel.

```
aws ecs list-account-settings
```

Exemple de sortie

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "on",
        "principalArn": "arn:aws:iam::123456789012:doej",
        "type": aws-managed"
    }
}
```

VPC IPv6 à double pile

Amazon ECS prend en charge la fourniture de tâches avec une adresse IPv6 en plus de l'adresse IPv4 privée principale.

Pour que les tâches reçoivent une adresse IPv6, la tâche doit utiliser le mode de réseau `awsvpc`, doit être lancé dans un VPC configuré pour le mode de double pile et le paramètre de compte `dualStackIPv6` doit être activé. Pour plus d'informations sur les autres exigences, consultez les sections Utilisation d'un VPC en mode double pile pour la capacité EC2, Utilisation d'un VPC en mode double pile pour la capacité des instances gérées Amazon ECS et Utilisation d'un VPC en mode double pile pour la capacité Fargate.

> **Important**
>
> Le paramètre de compte `dualStackIPv6` ne peut être modifié qu'à l'aide de l'API Amazon ECS ou de la AWS CLI. Pour de plus amples informations, veuillez consulter Modification des paramètres de compte Amazon ECS.

Si vous aviez une tâche en cours d'exécution avec le mode réseau `awsvpc` dans un sous-réseau IPv6 activé entre le 1er octobre 2020 et le 2 novembre 2020, le paramètre de compte par défaut `dualStackIPv6` dans la région dans laquelle la tâche était en cours d'exécution est `disabled`. Si cette condition n'est pas remplie, le paramètre de compte par défaut `dualStackIPv6` dans la région est `enabled`.

La valeur par défaut est `disabled`.

Mode de pilote du journal par défaut

Amazon ECS prend en charge la définition d'un mode de livraison par défaut des messages de journal depuis un conteneur vers le pilote de journal choisi. Le mode de livraison affecte la stabilité de l'application lorsque le flux de journaux depuis le conteneur vers le pilote de journal est interrompu.

Le paramètre `defaultLogDriverMode` prend en charge deux valeurs : `blocking` et `non-blocking`. Pour plus d'informations sur ces modes de livraison, consultez le LogConfiguration manuel Amazon Elastic Container Service API Reference.

Si vous ne spécifiez aucun mode de livraison dans la `logConfiguration` de votre définition conteneur, le mode que vous spécifiez à l'aide de ce paramètre de compte sera utilisé par défaut.

Le mode de livraison par défaut est `non-blocking`.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode par défaut du pilote de journalisation, passant de `blocking` à `non-blocking` afin de privilégier la disponibilité des tâches plutôt que la journalisation. Pour continuer à utiliser le mode `blocking` après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'option `mode` dans votre définition de conteneur `logConfiguration` sur `blocking`.
> - Réglez le paramètre du compte `defaultLogDriverMode` sur `blocking`.

Pour définir un mode de pilote de journal par défaut sur `blocking`, vous pouvez exécuter la commande suivante.

```
aws ecs put-account-setting-default --name defaultLogDriverMode --value "blocking"
```

Affichage de vos paramètres de compte Amazon ECS à l'aide de la console

Affichez les paramètres de votre compte dans la console pour déterminer les fonctionnalités auxquelles vous avez accès.

> **Important**
>
> Les paramètres`dualStackIPv6`,`fargateFIPSMode`, `fargateTaskRetirementWaitPeriod` et le `fargateEventWindows` compte ne peuvent être consultés ou modifiés qu'à l'aide du AWS CLI.

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.
3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).

Modification des paramètres de compte Amazon ECS

Modifiez les paramètres de votre compte pour accéder aux fonctionnalités d'Amazon ECS.

Le `guardDutyActivate` paramètre est en lecture seule dans Amazon ECS et indique si la surveillance de l'exécution est activée ou désactivée par votre administrateur de sécurité dans votre compte Amazon ECS. GuardDuty contrôle les paramètres de ce compte en votre nom. Pour plus d'informations, consultez la section Protection des charges de travail Amazon ECS à l'aide de la surveillance d'exécution.

> **Important**
>
> Les paramètres`dualStackIPv6`,`fargateFIPSMode`, `fargateTaskRetirementWaitPeriod` et le `fargateEventWindows` compte ne peuvent être consultés ou modifiés qu'à l'aide du AWS CLI.

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.

3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).
4. Choisissez Mettre à jour.
5. (Facultatif) Pour augmenter ou diminuer le nombre de tâches que vous pouvez exécuter en mode réseau awsvpc pour chaque instance EC2, sous Jonction AWSVPC, sélectionnez Jonction AWSVPC.
6. (Facultatif) Pour utiliser ou arrêter d'utiliser CloudWatch Container Insights par défaut pour les clusters, sous Observabilité de CloudWatch Container Insights, choisissez l'une des options suivantes :

   - Pour utiliser Container Insights avec observabilité améliorée, choisissez Container Insights avec observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
   - Pour arrêter d’utiliser Container Insights, choisissez Désactivé.
7. Pour activer ou désactiver l’autorisation d’étiquetage, sous Autorisation d’étiquetage des ressources, sélectionnez ou désélectionnez Autorisation d’étiquetage des ressources.
8. (Facultatif) Pour configurer un mode pilote de journal par défaut lorsqu’aucun mode de livraison de journal n’est défini dans la `logConfiguration` d’un conteneur sous Mode pilote de journal par défaut, choisissez l’une des options suivantes :

   - Pour définir le mode pilote de journal par défaut sur `blocking`, choisissez Blocage.
   - Pour définir le mode du pilote de journal par défaut sur`non-blocking`, choisissez Non-blocking.
9. Sélectionnez Enregistrer les modifications.
10. Dans l'écran de confirmation, choisissez Confirm (Confirmer) pour enregistrer la sélection.

Étapes suivantes

Si vous avez activé Container Insights avec observabilité améliorée ou Container Insights, vous pouvez éventuellement mettre à jour vos clusters existants pour utiliser cette fonctionnalité. Pour de plus amples informations, veuillez consulter Mise à jour d’un cluster Amazon ECS..

Restauration des paramètres par défaut du compte Amazon ECS

Vous pouvez utiliser le Console de gestion AWS pour rétablir les paramètres par défaut de votre compte Amazon ECS.

L'option Revert to account default (Restaurer la valeur par défaut du compte) n'est disponible que lorsque les paramètres de votre compte ne sont plus les paramètres par défaut.

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation en haut de l'écran, sélectionnez la région pour laquelle vous souhaitez afficher vos paramètres de compte.
3. Dans la page de navigation, choisissez Account Settings (Paramètres du compte).
4. Choisissez Mettre à jour.
5. Choisissez Revert to account default (Restaurer la valeur par défaut du compte).
6. Dans l'écran de confirmation, choisissez Confirm (Confirmer) pour enregistrer la sélection.

Gérer les paramètres du compte Amazon ECS à l'aide du AWS CLI

Vous pouvez gérer les paramètres de votre compte à l'aide de l'API Amazon ECS AWS CLI ou des kits de développement logiciel (SDK). Les paramètres`dualStackIPv6`,`fargateFIPSMode`, `fargateTaskRetirementWaitPeriod` et les paramètres du `fargateEventWindows` compte ne peuvent être consultés ou modifiés qu'à l'aide de ces outils.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI kits SDK et de l'API Amazon ECS via IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Pour plus d'informations sur les actions d'API disponibles pour les définitions de tâches, veuillez consulter Actions relatives à la configuration du compte dans la Référence de l'API Amazon Elastic Container Service (langue française non garantie).

Utilisez l'une des commandes suivantes pour modifier les paramètres de compte par défaut pour tous les utilisateurs ou rôles de votre compte. Ces modifications s'appliquent à l'ensemble du AWS compte, à moins qu'un utilisateur ou un rôle ne remplace explicitement ces paramètres pour lui-même.

- put-account-setting-default (AWS CLI)

```
aws ecs put-account-setting-default --name serviceLongArnFormat --value enabled --
region us-east-2
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre name par le paramètre de compte correspondant.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSettingDefault -Name serviceLongArnFormat -Value enabled -Region us-
east-1 -Force
```

Pour modifier les paramètres de votre compte utilisateur (AWS CLI)

Utilisez l'une des commandes suivantes pour modifier les paramètres de compte pour votre utilisateur . Si vous utilisez ces commandes en tant qu'utilisateur root, ces modifications s'appliquent à l'ensemble du compte AWS , à moins qu'un utilisateur ou un rôle remplace explicitement ces paramètres pour lui-même.

- put-accourt-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --region us-
east-1
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre name par le paramètre de compte correspondant.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -Force
```

Pour modifier les paramètres du compte pour un utilisateur ou un rôle spécifique (AWS CLI)

Utilisez l'une des commandes suivantes et spécifiez dans la requête l'ARN d'un utilisateur, d'un rôle ou de l'utilisateur root pour modifier les paramètres de compte pour un utilisateur ou un rôle spécifique.

- put-accourt-setting (AWS CLI)

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --principal-
arn arn:aws:iam::aws_account_id:user/principalName --region us-east-1
```

Vous pouvez également utiliser cette commande pour modifier d'autres paramètres de compte. Pour ce faire, remplacez le paramètre `name` par le paramètre de compte correspondant.

- Write-ECSAccountSetting (AWS Tools for Windows PowerShell)

```
Write-ECSAccountSetting -Name serviceLongArnFormat -Value enabled -PrincipalArn
 arn:aws:iam::aws_account_id:user/principalName -Region us-east-1 -Force
```

Rôles IAM pour Amazon ECS

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Utilisez le tableau suivant pour déterminer quels rôles IAM vous avez besoin pour Amazon ECS.

Role	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargate ou sur une instance externe et : • extrait une image de conteneur depuis un référentiel privé Amazon ECR. • extrait une image de conteneur à partir d'un référentiel privé	Rôle IAM d'exécution de tâche Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
		Amazon ECR dans un compte différent de celui qui exécute la tâche ; • envoie les journaux des conteneurs à CloudWatch Logs à l'aide du pilote de `awslogs` journal. Votre tâche est hébergée sur l'une AWS Fargate ou l'autre des instances Amazon EC2 et : • utilise l’authent ification de registre privé ; • utilise la surveilla nce d’exécution ; • la définition de tâche fait référence à des données sensibles à l'aide des secrets de Secrets Manager ou AWS des paramètres du magasin de paramètres de Systems Manager.	

Role	Définition	En cas de besoin	En savoir plus
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS services, tels qu'Amazon S3.	rôle IAM de tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos instances EC2 ou à vos instances externes de s’enregistrer auprès du cluster.	Votre tâche est hébergée sur des instances Amazon EC2 ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS
Rôle Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'accéder aux AWS API.	Votre tâche est hébergée sur des instances externes.	Rôle IAM Amazon ECS Anywhere
Rôles d’infrastructure Amazon ECS pour les équilibreurs de charge	Ce rôle permet à Amazon ECS de gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom pour les blue/green déploiements.	Vous souhaitez utiliser les blue/green déploiements Amazon ECS.	Rôle IAM d’infrastructure Amazon ECS pour les équilibreurs de charge
CodeDeploy Rôle Amazon ECS	Ce rôle permet CodeDeploy de mettre à jour vos services.	Vous utilisez le type CodeDeploy blue/green de déploiement pour déployer des services.	Amazon ECS CodeDeploy Rôle IAM
EventBridge Rôle Amazon ECS	Ce rôle permet EventBridge de mettre à jour vos services.	Vous utilisez les EventBridge règles et les objectifs pour planifier vos tâches.	Amazon ECS EventBridge Rôle IAM

Role	Définition	En cas de besoin	En savoir plus
rôle d'infrastructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters.	• Vous souhaitez utiliser les instances gérées Amazon ECS pour vos capacités. • Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS dont le type de lancement est Fargate ou EC2. Le rôle d'infrastructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM d'infrastructure Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
Profil d’instance	Ce rôle permet aux instances gérées Amazon ECS d’assumer le rôle d’infrastructure en toute sécurité.	Vous utilisez des instances gérées Amazon ECS dans vos clusters.	Profil d’instance des instances gérées Amazon ECS

Architecture des instances gérées Amazon ECS

Les instances gérées Amazon ECS sont une option de calcul entièrement gérée pour Amazon ECS qui vous permet d'exécuter des charges de travail conteneurisées sur la gamme complète des types d'instances Amazon EC2 tout en confiant la gestion de l'infrastructure à AWS. Avec Amazon ECS Managed Instances, vous pouvez accéder à des fonctionnalités de calcul spécifiques telles que l'accélération GPU, des architectures de processeur spécifiques, des performances réseau élevées et des types d'instances spécialisés, tout en AWS gère le provisionnement, la mise à l'échelle, l'application de correctifs et la maintenance de l'infrastructure sous-jacente.

Lorsque vous utilisez des instances gérées Amazon ECS, vous conditionnez votre application dans des conteneurs et vous spécifiez vos besoins de calcul. AWS sélectionne automatiquement les types d'instances Amazon EC2 à usage général les plus optimisés en termes de coûts qui répondent à vos besoins en matière de charge de travail, ou vous pouvez spécifier les attributs d'instance souhaités, notamment les types d'instance, les fabricants de processeurs et les accélérateurs. Les instances gérées Amazon ECS gèrent entièrement tous les aspects de l'infrastructure, notamment la mise à l'échelle, l'application de correctifs et l'optimisation des coûts, sans compromettre l'accès à AWS fonctionnalités et intégrations Amazon EC2.

Les instances gérées Amazon ECS prennent en charge les conteneurs Linux avec des optimisations et des configurations de sécurité spécifiques à la plateforme. Par défaut, les instances gérées Amazon ECS optimisent l'utilisation de l'infrastructure en plaçant plusieurs petites tâches sur des instances plus grandes, ce qui contribue à réduire les coûts et à améliorer les délais de lancement des tâches.

Cette rubrique décrit les différents composants des tâches et services d'instances gérées Amazon ECS, et souligne les considérations particulières relatives à l'utilisation des instances gérées Amazon ECS avec Amazon ECS.

Prise en main

Pour commencer à utiliser les instances gérées Amazon ECS, vous devez créer les rôles IAM requis et activer les instances gérées Amazon ECS dans votre AWS compte. Vous pouvez ensuite créer un fournisseur de capacité et lancer des tâches ou des services à l'aide du fournisseur de capacité d'instances gérées Amazon ECS.

Pour obtenir des instructions détaillées expliquant comment démarrer, consultez la section :

- Création d’une tâche pour les instances gérées Amazon ECS
- Apprenez à créer une tâche pour les instances gérées Amazon ECS à l'aide du AWS CLI

Fournisseurs de capacité

Les instances gérées Amazon ECS font appel à des fournisseurs de capacité pour gérer la capacité de calcul de vos charges de travail. Lorsque vous créez un fournisseur de capacité, vous pouvez utiliser la sélection d'instance par défaut ou spécifier des exigences d'instance personnalisées à l'aide de `instanceRequirements`.

Les options suivantes sont disponibles pour les fournisseurs de capacité :

- Fournisseur de capacité par défaut : sélectionne automatiquement les types d’instances à usage général les plus optimisés en termes de coûts en fonction de vos besoins en matière de charge de travail.
- Fournisseurs de capacité personnalisés : vous permettent de spécifier les attributs des instances à l’aide d’une sélection de type d’instance basée sur les attributs, notamment le nombre de vCPU, la mémoire, les fabricants d’UC, les types d’accélérateurs et les types d’instances spécifiques.

Une stratégie de fournisseur de capacité ne peut contenir qu’un seul type de fournisseur de capacité dans la liste suivante :

- Instances gérées Amazon ECS
- Groupe Auto Scaling
- Fargate/Fargate_ENDROIT

Sélection et optimisation des instances

Amazon ECS choisit les types d’instances pour les charges de travail de vos instances gérées Amazon ECS en utilisant l’une des méthodes suivantes :

- Sélection automatique : lorsque vous utilisez le fournisseur de capacité par défaut, Amazon ECS sélectionne automatiquement les types d’instances à usage général les plus optimisés en termes de coûts qui répondent aux exigences en matière d’UC et de mémoire spécifiées dans votre définition de tâche.

- Attribute-based sélection - Lorsque vous utilisez des fournisseurs de capacité personnalisés, vous pouvez spécifier des attributs d'instance tels que le nombre de processeurs virtuels, la taille de la mémoire, les fabricants de processeurs, les types d'accélérateurs et des types d'instances spécifiques. Amazon ECS effectue sa sélection parmi tous les types d'instances qui correspondent aux attributs que vous avez spécifiés.

Les instances gérées Amazon ECS optimisent l'utilisation et les coûts de l'infrastructure par le biais de plusieurs mécanismes :

- Multi-task placement : par défaut, Amazon ECS place plusieurs petites tâches sur des instances plus importantes afin d'optimiser l'utilisation et de réduire les coûts.
- Consolidation active de la charge de travail : Amazon ECS identifie les cas où les instances de conteneur sont réellement inactives tout en essayant d'éviter une interruption prématurée susceptible d'avoir un impact sur la disponibilité des applications ou les performances de déploiement. Le système respecte le nombre minimum et maximum de tâches définis pour un service, le comportement de démarrage avant arrêt et le comportement de protection des tâches.
- Right-sizing - À mesure que les exigences en matière de charge de travail évoluent, Amazon ECS lance des instances de remplacement dont la taille est adaptée aux besoins actuels.

Amazon ECS utilise les fenêtres d'événements Amazon EC2 pour planifier les activités de maintenance aux périodes de votre choix. Les fenêtres d'événement vous permettent de définir des périodes récurrentes pendant lesquelles AWS peut effectuer la maintenance de vos instances, ce qui vous aide à minimiser les perturbations de vos charges de travail en alignant la maintenance sur votre calendrier opérationnel. Pour plus d'informations, consultez la section Événements planifiés pour vos instances dans le Guide de l'utilisateur Amazon EC2.

Si vous avez besoin d'une isolation renforcée, vous pouvez configurer les instances gérées Amazon ECS pour exécuter chaque tâche sur une instance distincte avec VM-level limites d'isolation de sécurité.

Définitions de tâche

Les tâches qui utilisent des instances gérées Amazon ECS prennent en charge la plupart des paramètres de définition de tâches Amazon ECS. Les instances gérées Amazon ECS sont compatibles avec les définitions de tâches Fargate existantes à l'aide de la version de plateforme 1.4.0, ce qui simplifie la migration.

Pour utiliser les instances gérées Amazon ECS, définissez le paramètre de définition de tâche `requiresCompatibilities` pour qu'il contienne `MANAGED_INSTANCES`. Vos définitions de tâches peuvent spécifier la compatibilité des instances gérées par Fargate et Amazon ECS afin de garantir la flexibilité des options de déploiement.

Système d'exploitation et architecture de l'UC

Les systèmes d'exploitation suivants sont pris en charge :

- Fusée à bouteilles

Deux architectures sont disponibles pour la définition de tâche Amazon ECS, ARM et X86_64.

Lorsque vous exécutez des conteneurs Linux sur des instances gérées Amazon ECS, vous pouvez utiliser l'architecture CPU X86_64 ou l'architecture ARM64 pour votre ARM-based candidatures.

Fonctionnalités principales

Voici les principales fonctionnalités des instances gérées Amazon ECS :

- Sélectionnez des types d'instances EC2 spécifiques pour répondre aux exigences de votre application, en permettant d'accéder à des fonctionnalités matérielles spécialisées telles que GPU-accelerated calcul, capacités spécifiques du processeur et grandes tailles de mémoire.
- Optimisation de l'utilisation des ressources et les coûts avec plusieurs tâches sur une seule instance par défaut, contrairement à Fargate qui exécute chaque tâche dans son propre environnement isolé.
- Garantissez la conformité en matière de sécurité et l'application régulière de correctifs aux instances. ECS Managed Instances lance la vidange des instances au bout de 14 jours et remplace automatiquement les tâches basées sur les services par de nouvelles instances.
- Activez des fonctions avancées de mise en réseau et d'administration système dans des conteneurs à l'aide de fonctionnalités Linux privilégiées, notamment CAP_NET_ADMIN, CAP_SYS_ADMIN, CAP_BPF et CAP_PERFMON.

Rôles IAM

Les instances gérées Amazon ECS nécessitent deux rôles IAM :

- Rôle de l'infrastructure: Ce rôle permet AWS pour gérer les instances gérées Amazon ECS en votre nom.
- Profil d’instance : un profil d’instance est un moyen de transmettre un rôle IAM aux instances gérées Amazon ECS. Ce profil est utilisé pour :
  - Définissez les autorisations IAM pour les instances gérées Amazon ECS qui exécutent vos charges de travail de conteneur.
  - Autoriser AWS pour gérer ces instances en votre nom.
  - Autoriser l'accès aux instances AWS services en fonction des autorisations définies dans le profil.

Conformité et sécurité

Les instances gérées Amazon ECS mettent en œuvre plusieurs niveaux de sécurité pour protéger vos charges de travail :

- Configuration sécurisée - Les instances gérées Amazon ECS suivent AWS les meilleures pratiques de sécurité, notamment l'absence d'accès SSH, un système de fichiers racine immuable et des contrôles d'accès obligatoires au niveau du noyau via SELinux.
- Patching automatique - AWS met régulièrement à jour les instances gérées Amazon ECS avec les derniers correctifs de sécurité, en respectant les fenêtres de maintenance que vous configurez.
- Durée de vie limitée de l'instance - ECS lance automatiquement la vidange des instances au bout de 14 jours, garantissant ainsi que vos applications s'exécutent sur des instances correctement configurées avec des correctifs de sécurité à jour.
- Fonctionnalités privilégiées : vous pouvez éventuellement activer des fonctionnalités Linux privilégiées pour les charges de travail qui en ont besoin, telles que les solutions de surveillance et d’observabilité du réseau.

Les instances gérées Amazon ECS prennent en charge les mêmes programmes de conformité qu'Amazon ECS, notamment PCI-DSS, HIPAA et FedRAMP. Dans les régions prises en charge, vous pouvez configurer les points de terminaison FIPS au niveau du fournisseur de capacité afin de vous conformer à FedRAMP.

Mise en réseau

Les instances gérées Amazon ECS prennent en charge les modes réseau `awsvpc` et `host`. Le mode réseau `awsvpc` fournit à chaque tâche sa propre interface réseau Elastic et sa propre adresse IP privée au sein de votre VPC. Cela permet des contrôles ACL précis des groupes de sécurité et du réseau au niveau des tâches. En mode réseau `host`, les tâches partagent l'espace de noms réseau de l'instance gérée Amazon ECS hôte. Pour plus d'informations sur la mise en réseau des tâches sur les instances gérées Amazon ECS, consultez la section Mise en réseau des tâches Amazon ECS pour les instances gérées Amazon ECS.

Stockage d'instances

Les instances gérées Amazon ECS prennent en charge la configuration de la taille du volume de données Amazon EBS attaché à l'instance. Ce stockage est partagé entre toutes les tâches qui s'exécutent sur l'instance et peut être utilisé pour les montages liés (bind mounts). Le volume peut être monté et partagé entre les conteneurs qui utilisent les paramètres `volumes`, `mountPoint` et `volumesFrom` dans la définition de tâche.

Le volume est attaché lors de la création de l'instance. Vous pouvez spécifier la taille du volume, en Gio, lorsque vous créez un fournisseur de capacité d'instances gérées Amazon ECS à l'aide du paramètre `storageConfiguration`.

```
{
...

    "managedInstancesProvider": {
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/
ecsInstanceProfile",
            "networkConfiguration": {
                "subnets": [
                    "subnet-abcdef01234567",
                    "subnet-bcdefa98765432"
                ],
                "securityGroups": [
                    "sg-0123456789abcdef"
                ]
            },
```

```
            "storageConfiguration": {
                "storageSizeinGiB" : 100

            }
        }
    }
 ...
}
```

La taille minimale de ce volume est de 30 Gio et la taille maximale est de 16 384 Gio. Par défaut, la taille de ce volume est de 80 Gio.

L'image du conteneur extraite, compressée et non compressée pour la tâche est stockée dans le volume. Pour déterminer la quantité totale de stockage d'instance que votre tâche doit utiliser comme montage lié, vous devez soustraire la quantité de stockage utilisée par votre image de conteneur de la quantité totale de stockage d'instance allouée à votre tâche.

Les performances des volumes Amazon EBS attachés aux instances gérées Amazon ECS correspondent aux performances des instances Amazon EC2 correspondantes, comme indiqué dans le Amazon EBS-optimized les instances documentation dans le Guide de l'utilisateur d'Amazon EC2.

Vous pouvez créer des instantanés du volume afin d'effectuer une analyse approfondie des problèmes de sécurité ou de déboguer votre application. Pour plus d'informations sur la création d'instantanés de volumes Amazon EBS, consultez la section Instantanés Amazon EBS dans le Guide de l'utilisateur Amazon EBS. Si le chiffrement Amazon EBS est activé par défaut, le volume sera chiffré avec AWS KMS clé spécifiée pour le chiffrement par défaut. Pour plus d'informations sur le chiffrement par défaut, consultez la section Activer le chiffrement Amazon EBS par défaut dans le Guide de l'utilisateur Amazon EBS.

Outre l'utilisation du volume de données attaché à l'instance, vous pouvez également configurer des volumes de données pour chaque tâche exécutée sur les instances gérées Amazon ECS. Pour plus d'informations sur les options de stockage disponibles au niveau des tâches, consultez la section Options de stockage pour les tâches Amazon ECS.

Stockage local

Magasin d'instances Amazon EC2 fournit un stockage temporaire au niveau des blocs pour vos instances Amazon EC2. Le stockage fourni par le magasin d'instance Amazon EC2 est accessible

via des disques physiquement connectés aux hôtes. Vous pouvez configurer les instances gérées Amazon ECS pour utiliser le stockage d'instance en tant que volume de données pour les tâches exécutées sur l'instance de conteneur. Lorsque vous activez le stockage local et que l'instance possède des volumes de stockage d'instance, Amazon ECS utilise les volumes de stockage d'instance au lieu de provisionner un volume de données Amazon EBS. Cela peut réduire les coûts de stockage et améliorer I/O performances pour les charges de travail sensibles à la latence. L'utilisation des volumes de stockage d'instances avec les instances gérées Amazon ECS est gratuite.

Lorsqu'une instance possède plusieurs volumes de stockage d'instance, Amazon ECS les combine automatiquement en un seul volume RAID 0 et le présente comme un stockage contigu pour les tâches. Lorsqu'une instance ne possède pas de volumes de stockage d'instance ou lorsque le stockage local est désactivé, Amazon ECS provisionne un volume de données Amazon EBS dont la taille est spécifiée dans `storageSizeGiB`. Les volumes de stockage d'instances sont éphémères. Les données des volumes de stockage d'instance sont perdues lorsque l'instance est arrêtée, arrêtée ou en cas de panne matérielle. N'utilisez pas le stockage d'instance pour les données qui doivent persister.

Pour activer le stockage local, créer un nouveau fournisseur de capacités avec le `localStorageConfiguration` paramètre et ensemble `useLocalStorage` pour `true`. Vous pouvez également utiliser le `instanceRequirements` paramètre pour garantir que les instances provisionnées incluent un stockage local d'une taille spécifique.

> **Note**
>
> Lorsque vous créez un cluster à l'aide du Console de gestion AWS, le fournisseur de capacité d'instances gérées par défaut créé par Amazon ECS n'inclut pas `localStorageConfiguration`.

L'exemple suivant montre une configuration de fournisseur de capacité avec le stockage local activé et des exigences d'instance qui spécifient SSD-based stockage d'instance avec un minimum de 50 GiB.

```
{
...
    "managedInstancesProvider": {
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
```

```
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/
ecsInstanceProfile",
            "networkConfiguration": {
                "subnets": [
                    "subnet-abcdef01234567",
                    "subnet-bcdefa98765432"
                ],
                "securityGroups": [
                    "sg-0123456789abcdef"
                ]
            },
            "storageConfiguration": {
                "storageSizeGiB": 50
            },
            "localStorageConfiguration": {
                "useLocalStorage": true
            },
            "instanceRequirements": {
                "vCpuCount": {
                    "min": 0
                },
                "memoryMiB": {
                    "min": 0
                },
                "localStorage": "REQUIRED",
                "totalLocalStorageGB": {
                    "min": 50
                },
                "localStorageTypes": ["ssd"]
            }
        }
    }
}
```

Lorsque vous omettez `instanceRequirements` configuration, paramètres par défaut d'Amazon ECS `localStorage` pour `INCLUDED`. Dans ce cas, Amazon ECS prend en compte les instances avec et sans volumes de stockage d'instance pour le provisionnement. Les instances dotées de volumes de stockage d'instance utilisent un magasin d'instance pour le volume de données, tandis que les instances sans volumes de stockage d'instance utilisent un volume de données Amazon EBS. Quand `totalLocalStorageGB` n'est pas spécifié, Amazon ECS utilise tout le stockage d'instance disponible, sans qu'aucune taille minimale ne soit imposée.

Le `storageSizeGiB` valeur en `storageConfiguration` définit uniquement la taille du volume de données Amazon EBS et est utilisé lorsque le stockage d'instance n'est pas disponible. Le `totalLocalStorageGB` valeur en `instanceRequirements` contrôle la taille de stockage minimale requise pour les instances provisionnées. Pour garantir que toutes les instances utilisent le stockage local, définissez `localStorage` pour `REQUIRED` et spécifiez une taille minimale en utilisant `totalLocalStorageGB`.

Pour vérifier si une instance de conteneur provisionnée spécifique utilise le stockage local, vérifiez `ecs.capability.storage.local-storage-enabled` attribut sur l'instance de conteneur.

Équilibrage de charge des services

Vos services Amazon ECS utilisant des instances gérées Amazon ECS peuvent être configurés pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches de votre service.

Les services Amazon ECS sur instances gérées Amazon ECS prennent en charge les types d'équilibreurs de charge Application Load Balancer, Network Load Balancer et Gateway Load Balancer. Itinéraire des équilibreurs de charge des applications HTTP/HTTPS le trafic (couche 7), tandis que les équilibreurs de charge réseau acheminent le trafic TCP ou UDP (couche 4).

De même, lorsque vous créez des groupes cible pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches utilisant le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.

Surveillance et observabilité

Amazon ECS Managed Instances fournit des fonctionnalités de surveillance complètes via CloudWatch mesures et intégration avec les outils d'observabilité :

- CloudWatch indicateurs - Surveillez l'utilisation du processeur, de la mémoire, du réseau et du stockage à la fois au niveau de la tâche et de l'instance.
- Container Insights : obtenez des métriques de performance et des journaux détaillés pour vos applications conteneurisées.
- Third-party intégrations - Lorsque les fonctionnalités privilégiées sont activées, vous pouvez exécuter des solutions avancées de surveillance et d'observabilité qui nécessitent des autorisations Linux élevées.

Tarification et optimisation des coûts

Avec les instances gérées Amazon ECS, l'intégralité de l'instance Amazon EC2 qui exécute vos tâches vous est facturée. La tarification dépend des types d'instances sélectionnés pour vos charges de travail.

Les instances gérées Amazon ECS proposent plusieurs fonctionnalités d'optimisation des coûts :

- Multi-task optimisation - Optimisez l'utilisation des instances en exécutant plusieurs tâches sur des instances de taille appropriée.

Vos Savings Plans Compute and Instance s'appliquent également aux charges de travail des instances gérées Amazon ECS.

Quotas de service

Les charges de travail des instances gérées Amazon ECS sont soumises à votre Amazon EC2 On-Demand quotas de service d'instance. Vos services Amazon ECS utilisant des instances gérées Amazon ECS sont soumis à des quotas de service Amazon ECS.

Pour de plus amples informations sur les quotas de service, consultez la section :

- Quotas et points de terminaisons Amazon EC2 dans le Référence générale d'Amazon Web Services
- Quotas de service Amazon ECS service

Considérations concernant la migration

La migration vers Amazon ECS Managed Instances est directe pour la plupart des charges de travail :

- Depuis Fargate : ne nécessite qu'une modification de la configuration du fournisseur de capacité et un redéploiement. Les définitions de tâches existantes utilisant la version de plateforme 1.4.0 sont entièrement compatibles.
- Depuis EC2 : similaire à la migration vers Fargate, mais vous conservez l'accès aux fonctionnalités d'Amazon EC2, telles que des types d'instances spécifiques.

Tenez compte des éléments suivants lorsque vous planifiez votre migration :

- Les applications doivent tolérer la durée de vie de l'instance de 14 jours et les fenêtres de maintenance planifiées.
- Long-running les tâches (supérieures à 14 jours) ne sont pas adaptées aux instances gérées Amazon ECS.
- Les AMI personnalisées ne sont pas prises en charge. Les instances gérées Amazon ECS utilisent AWS-AMI gérées et optimisées pour la sécurité.

Limites et considérations

Les restrictions suivantes s'appliquent aux instances gérées Amazon ECS :

- AMI personnalisées - L'AMI est détenue et gérée par AWS
- Durée de vie de l'instance : durée d'exécution maximale de 14 jours par instance pour garantir les correctifs de sécurité et la conformité.
- Accès SSH : non disponible pour des raisons de sécurité. Utilisation d'Amazon ECS Exec pour le débogage et le dépannage. Opérations de gestion via les API Amazon ECS uniquement.

Contrôles de l'organisation

Certains contrôles organisationnels peuvent empêcher les instances gérées Amazon ECS de fonctionner correctement. Si tel est le cas, vous devez mettre à jour ces contrôles pour permettre à Amazon ECS de disposer des autorisations requises pour gérer les instances EC2 en votre nom.

Amazon ECS utilise un rôle d'infrastructure pour lancer les instances EC2 qui soutiennent les instances gérées Amazon ECS. Ce rôle d'infrastructure est Rôle IAM qui est créé sur votre compte et permet à Amazon ECS de gérer les instances gérées Amazon ECS en votre nom. Politiques de contrôle des services (SCP) s'appliquent toujours aux actions effectuées avec des rôles d'infrastructure. Cela permet à un SCP d'inhiber les opérations des instances gérées Amazon ECS. Le cas le plus courant survient lorsqu'une SCP restreint les images Amazon Machine Image (AMI) pouvant être lancées. Pour permettre aux instances gérées Amazon ECS de fonctionner, modifiez le SCP pour autoriser le lancement d'AMI à partir de comptes AMI Amazon ECS Managed Instances.

Vous pouvez également utiliser la fonctionnalité EC2 Allowed AMIs afin de limiter la visibilité des AMI provenant d'autres comptes. Si vous utilisez cette fonctionnalité, vous devez étendre les critères

d'image pour inclure également les comptes AMI Amazon ECS Managed Instances dans les régions qui vous intéressent.

Exemple de SCP pour bloquer toutes les AMI à l'exception des AMI Amazon ECS Managed Instances

Le SCP ci-dessous empêche les appels `ec2:RunInstances` sauf si l'AMI appartient au compte AMI Amazon ECS Managed Instances pour us-west-2 ou us-east-1.

> **Note**
>
> C'est important pas pour utiliser le `ec2:Owner` clé de contexte. Amazon est propriétaire des comptes AMI Amazon ECS Managed Instances et la valeur de cette clé sera toujours `amazon`. Construction d'un SCP qui permet de lancer des AMI si `ec2:Owner` est `amazon` permettra de lancer n'importe quelle AMI appartenant à Amazon, et pas uniquement celles destinées aux instances gérées Amazon ECS.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "DenyAMI",
      "Effect": "Deny",
      "Action": "ec2:RunInstances",
      "Resource": "arn:*:ec2:*::image/ami-*",
      "Condition": {
        "StringNotEquals": {
          "aws:ResourceAccount": [
            "187296253231",
            "260073348889"
          ]
        }
      }
    }
  ]
}
```

Comptes AMI Amazon ECS Managed Instances

AWS les comptes qui varient d'une région à l'autre hébergent des AMI publiques Amazon ECS Managed Instances.

AWS Région	Compte
af-south-1	070957084703
ap-east-1	587573215167
ap-northeast-1	679336465495
ap-northeast-2	309903600357
ap-northeast-3	384570461223
ap-south-1	062344138989
ap-south-2	624198668379
ap-southeast-1	832199679391
ap-southeast-2	552073033681
ap-southeast-3	368903466070
ap-southeast-4	696793786439
ap-southeast-5	003457290689
ap-southeast-6	465836752572
ap-southeast-7	622515864387
ca-central-1	853167153192
ca-west-1	899469777611
eu-central-1	832570432258
eu-central-2	041659148495

AWS Région	Compte
eu-north-1	851563870067
eu-south-1	766433696616
eu-south-2	003380494496
eu-west-1	986619735082
eu-west-2	591706807364
eu-west-3	108582616801
il-central-1	009537862704
me-central-1	540883425316
me-south-1	181438624895
mx-central-1	210749644920
sa-east-1	591338347621
us-east-1	260073348889
us-east-2	292185169523
us-west-1	187296253231
us-west-2	491085424538

Types d’instances gérées Amazon ECS

Lorsque vous créez un fournisseur de capacité Amazon ECS Managed Instances, vous choisissez la manière dont Amazon ECS sélectionne les types d'instances EC2 pour vos tâches. Vous pouvez laisser Amazon ECS choisir automatiquement, spécifier des types ou des familles d'instances exacts, ou définir les attributs matériels auxquels les instances doivent répondre.

Méthodes de sélection d’instances

Les instances gérées Amazon ECS proposent deux méthodes pour sélectionner les types d’instances :

- Type ou famille d'instances spécifiques: vous spécifiez des types d'instances exacts (tels que `m6i.xlarge`) ou des familles d'instances utilisant des caractères génériques (comme `c6i.*` ou `m6*.*`). Utiliser `allowedInstanceTypes` pour inclure des types spécifiques, ou `excludedInstanceTypes` pour exclure les types que vous ne souhaitez pas.
- Attribute-based sélection: vous définissez les exigences matérielles (telles que le nombre de processeurs virtuels, la mémoire, le fabricant du processeur ou le type d'accélérateur) et Amazon ECS sélectionne les types d'instance qui correspondent à tous les attributs que vous avez spécifiés.

Si vous ne spécifiez aucun type d'instance ni aucun attribut, Amazon ECS sélectionne automatiquement les types d'instance les plus optimisés en termes de coûts en fonction de vos exigences en matière de définition de tâches.

Sélection de type d’instance spécifique

En sélectionnant un type d'instance spécifique, vous contrôlez exactement quels types d'instances EC2 Amazon ECS Managed Instances peuvent utiliser. Vous pouvez spécifier des types d'instances individuels, des familles d'instances complètes ou une combinaison des deux.

Utilisez le `allowedInstanceTypes` champ pour spécifier les types à inclure, ou le `excludedInstanceTypes` champ pour exclure les types que vous ne souhaitez pas. Les deux champs acceptent les caractères génériques :

- `m6i.xlarge` — un type d'instance spécifique
- `c6i.*` — toutes les tailles de la famille C6i
- `m6*.*` — toutes les familles de générations m6 (m6i, m6a, m6g, etc.)

Attribute-based sélection du type d'instance

Avec la sélection du type d'instance basée sur les attributs, vous définissez les exigences matérielles et Amazon ECS sélectionne les types d'instance qui répondent à ces exigences. Cela apporte de

la flexibilité et permet de garantir que vos tâches sont correctement placées, même si des types d'instances spécifiques ne sont pas disponibles.

Lorsque vous spécifiez plusieurs attributs, Amazon ECS sélectionne des types d'instance qui répondent à tous les attributs spécifiés. Si vous spécifiez plusieurs valeurs pour un attribut, Amazon ECS sélectionne les types d'instance qui répondent à toutes les valeurs spécifiées.

Vous pouvez filtrer par attributs dans les catégories suivantes :

- CPU et mémoire — nombre de processeurs virtuels, mémoire (MiB), ratio mémoire/processeur virtuel, fabricants de processeurs (Intel, AMD ou AWS Graviton) et génération d'instances (actuelle ou précédente).
- Stockage local — Incluez, excluez ou exigez un stockage local. Filtrez par type de stockage (disque dur ou SSD) et par capacité de stockage totale.
- Réseau et performance — Bande passante réseau (Gbit/s), nombre d'interfaces réseau, bande passante EBS de référence (Mbit/s), prise en charge des performances en rafale et instances bare metal.
- Accélérateurs — Nombre d'accélérateurs, fabricants, noms des puces, types d'accélérateurs et mémoire totale de l'accélérateur.
- Protection des prix — Prix spot maximum et On-Demand prix en pourcentage du type d'instance le moins cher qui correspond à vos besoins.

Pour obtenir la liste complète des attributs et des valeurs valides, voir InstanceRequirementsRequest dans le Référence de l'API Amazon ECS.

Options de facturation et d’achat

Vous choisissez un modèle de provisionnement lorsque vous créez un fournisseur de capacité. Le modèle de provisionnement détermine la manière dont Amazon ECS Managed Instances acquiert des instances EC2 pour vos tâches :

On-Demand

Payez la capacité de calcul à l’heure, sans engagement à long terme ni paiement initial. Il s’agit de l’option par défaut.

Spot

Vous utilisez la capacité EC2 disponible à un prix réduit. Les instances Spot peuvent être interrompues lorsque EC2 a besoin de récupérer de la capacité. Set `capacityOptionType` pour `Spot` lors de la création de votre fournisseur de capacité.

Réserve de capacité

Vous réservez de la capacité EC2 dans une zone de disponibilité spécifique. Set `capacityOptionType` pour `Reserved` et fournissez un groupe de réservation de capacité. Vous pouvez également spécifier une préférence de réservation : `reservations-only`, `reservations-first`, ou `reservations-excluded`.

Les instances gérées Amazon ECS bénéficient également automatiquement de AWS Plans d'épargne et instances réservées que vous avez achetés pour les types d'instances utilisés pour vos tâches. Aucune configuration supplémentaire n'est requise.

Types d'instances pris en charge par Amazon ECS Managed Instances

Les instances gérées Amazon ECS prennent en charge les types d'instances suivants.

- Optimisation du calcul : C5 | C5a | C5ad | C5d | C5n | C6a | C6g | C6gd | C6i | C6id | C6in | C6in | C7a | C7g | C7gD | C7gn | C7i | C7i-flex | C8a | C8g | C8 Go | C8Gd | C8Gn | C8i | C8iB | C8iD | C8i-flex | C8 pouces | C8 pouces
- Usage général : M5 | M5a | M5ad | M5d | M5dn | M5n | M5Zn | M6a | M6g | M6gD | M6i | M6id | M6IdN | M6 pouces | M7a | M7g | M7gD | M7i | M7i-flex | M8a | M8azn | M8g | M8 Go | M8Gd | M8GN | M8i | M8iB | M8ID | M8iDB | M8idn | M8i-flex | M8 pouces | M8 pouces | M9 g | M9 gD | T3 | T3 | T4
- Mémoire optimisée : R5 | R5a | R5ad | R5b | R5d | R5n | R6a | R6g | R6Gd | R6i | R6id | R6IdN | R6entrée | R7a | R7g | R7Gd | R7i | R7iz | R8a | R8g | R8Go | R8Gd | R8Gd | R8Gd | R8Gd GN | U-3tb1 | U7i-6tb | U7i-8tb | U7i-12tb | U7in-24tb | U7in-32tb | X2Gd | X2IDn | X2IEDN | X2ieZn | X8g | Z1d
- Optimisées pour le stockage : D3 | D3en | I3en | I4g | I4i | I7i | I7ie | I8g | I8ge | Im4gn | Is4gen
- Calcul accéléré : DL1 | G4ad | G4dn | G5 | G5 | G5 | G6 | G6e | G6f | G7e | Gr6 | Gr6f | Inf1 | Inf2 | P3dN | P4d | P4de | P5 | P5fr | P6-B200 | P6-B300 | Tr1
- High-performance informatique : HPC6a | HPC6ID | HPC7a | HPC7g

En outre, Amazon ECS Managed Instances crée uniquement des instances qui répondent aux exigences suivantes.

- Plus d'un processeur virtuel
- La taille de l'instance n'est pas nano ou micro

Pour plus d’informations, consultez la section Conventions de dénomination des types d’instances Amazon EC2.

Pratiques exemplaires en matière de sélection d’instances pour les instances gérées Amazon ECS

Il est essentiel de sélectionner la bonne configuration d’instance pour les charges de travail de vos instances gérées Amazon ECS afin d’optimiser les performances, les coûts et l’utilisation des ressources. Amazon ECS propose des options de sélection d’instances flexibles qui vous permettent de trouver un équilibre entre les exigences de votre application et la rentabilité. Les pratiques exemplaires suivantes vous aident à prendre des décisions éclairées concernant la sélection d’instances pour vos charges de travail conteneurisées.

1. Utiliser la sélection d'instance par défaut

   Lorsque vous ne spécifiez pas `instanceRequirements`, Amazon ECS choisit les instances les plus économiques qui répondent aux exigences suivantes en matière de définition de tâches et de paramètres de service :

   Définition de tâche
   - opérantSystemFamily
   - cpuArchitecture
   - cpu
   - memory

   Définition des services
   - placementConstraints
   - placementStrategy
2. Utiliser la sélection basée sur les attributs pour la plupart des charges de travail afin d’apporter de la flexibilité et d’améliorer les taux de réussite du placement

Attribute-based la sélection d'instances permet à Amazon ECS de choisir parmi une gamme plus large de types d'instances qui répondent à vos exigences spécifiques. Cette approche augmente les chances de réussite du placement des tâches et permet une meilleure optimisation des coûts en permettant à Amazon ECS de sélectionner les instances les plus économiques disponibles au moment du lancement.

3. Utiliser des types d’instances spécifiques uniquement lorsque les applications ont des exigences matérielles spécifiques

   Réservez une sélection de type d’instance spécifique pour les charges de travail qui nécessitent des fonctionnalités matérielles particulières, telles que l’accélération du GPU, des processeurs haute fréquence ou des capacités réseau spécialisées. Pour les applications générales, la sélection basée sur les attributs offre généralement une meilleure flexibilité et une meilleure optimisation des coûts.

4. Choisir des ressources équilibrées pour éviter la surallocation et les coûts inutiles

   Sélectionnez des configurations d’instance qui correspondent étroitement aux exigences en termes d’UC et de mémoire de votre application. Évitez une surallocation des ressources trop importante, car cela entraîne une augmentation des coûts et une réduction de l’efficacité. Utilisez les données de surveillance pour comprendre vos modèles réels d’utilisation des ressources et ajuster la sélection des instances en conséquence.

5. Combiner les types d’instances pour les applications dont les charges de travail sont différentes afin d’équilibrer les performances et les coûts

   Pour les applications présentant des exigences de performance ou des modèles de charge de travail variés, envisagez de faire appel à plusieurs fournisseurs de capacité avec différentes configurations d’instance. Cette approche vous permet d’optimiser les coûts en utilisant des types d’instances appropriés pour les différents composants de votre application tout en maintenant les performances là où cela est nécessaire.

6. Lorsque vous utilisez un fournisseur de capacité Amazon ECS Managed Instances configuré avec `capacityOptionType=Reserved`, sachez que les services ECS utilisent une configuration de déploiement par défaut de `minimumHealthyPercent=100%` et `maximumPercent=200%`, ce qui signifie que les déploiements ECS tentent de démarrer de nouvelles tâches avant d'arrêter les anciennes et nécessitent temporairement jusqu'à 200 % de votre capacité en régime permanent. Si votre service consomme toute la capacité disponible dans vos réservations de capacité EC2 à l'état stable, les déploiements échoueront car aucune capacité supplémentaire n'est disponible pour lancer de nouvelles tâches pendant le processus de déploiement. Pour éviter cela, réglez

`minimumHealthyPercent` à moins de 100 % (par exemple, 75 %) et envisagez de régler `maximumPercent` à 100 % pour garantir que le service arrête les tâches avant d'en démarrer de nouvelles, ce qui permet de mener à bien les déploiements en libérant de la capacité avant de lancer les tâches de remplacement. En outre, pensez à surveiller régulièrement l'utilisation de votre capacité afin de conserver une marge de manœuvre dans vos réservations, de gérer les déploiements et les pics de trafic.

Comportement d'extraction d'images de conteneurs Amazon ECS Managed Instances

Le temps de démarrage du conteneur dépend de la taille de l'image de conteneur sous-jacente. Les images plus grandes, telles que les versions complètes de Debian, Ubuntu ou Amazon Linux 2, mettent plus de temps à démarrer car elles incluent plus de services que leurs homologues plus compacts (Debian-slim, Ubuntu-slim, Amazon-slim) ou des images de base minimales telles que Alpine.

Lorsqu'Amazon ECS lance une nouvelle instance de conteneur Amazon ECS Managed Instances, le cache d'images local est vide. La première tâche exécutée sur cette instance de conteneur extrait l'image du conteneur du référentiel de conteneurs vers le cache d'images local. Les tâches suivantes qui utilisent la même image sur cette instance de conteneur utilisent l'image mise en cache au lieu de l'extraire à nouveau.

L'agent Amazon ECS sur les instances gérées Amazon ECS gère automatiquement l'espace disque sur chaque instance de conteneur. Lorsqu'une tâche s'arrête et qu'aucune autre tâche en cours d'exécution n'utilise la même image de conteneur, Amazon ECS lance un délai de nettoyage de 15 minutes pour cette image. Si aucune nouvelle tâche utilisant l'image ne démarre sur cette instance de conteneur dans les 15 minutes, l'image est supprimée du cache d'images local.

Résolution des images de conteneurs

Amazon ECS transforme les balises d'image de conteneur en résumés d'images afin de garantir que toutes les tâches d'un service exécutent la même version d'image. Ce comportement affecte le classement des instances gérées Amazon ECS et la sélection des instances, car Amazon ECS planifie d'abord une seule tâche à partir du service pour récupérer le résumé de l'image du conteneur avant de planifier les tâches restantes. Pour plus d’informations, consultez Résolution des images de conteneurs.

Application de correctifs dans les instances gérées Amazon ECS

Dans les instances gérées Amazon ECS, l'application de correctifs est un processus de maintenance essentiel dans lequel AWS gère et met à jour automatiquement les logiciels sur les instances de conteneurs gérées afin de garantir la sécurité et la conformité tout en maintenant la disponibilité de la charge de travail grâce à un processus contrôlé et configurable.

Cycle de vie d’une instance

Par défaut, les instances de conteneurs gérés Amazon ECS fonctionnent selon un cycle de vie normalisé de 14 à 21 jours. Amazon ECS lance un drainage progressif de la charge de travail au 14e jour suivant le lancement de l’instance, l’arrêt final ayant lieu au plus tard le 21e jour. Amazon ECS prend en charge le drainage anticipé dans des circonstances spécifiques :

- Détection de failles de sécurité sur le logiciel
- Dégradation de l’état du matériel
- Respect des fenêtres d’événements configurées par le client

Cette approche assure la conformité du système tout en respectant les exigences de maintenance définies par le client.

Fenêtres d’événements et planification de la maintenance

AWS gère le cycle de vie d'une instance de conteneur gérée grâce à des processus d'arrière-plan automatisés qui surveillent l'horodatage de création et les calendriers de maintenance d'un nœud. Lors du lancement de l'instance, AWS définit un calendrier de vidange par défaut sur 14 jours et évalue toutes les fenêtres d'événements configurées par le client.

Vous pouvez planifier des activités de maintenance pour les instances gérées Amazon ECS en configurant des fenêtres d'événements. Vous pouvez associer une ou plusieurs instances gérées Amazon ECS à une fenêtre d'événements à l'aide d'ID d'instance ou de balises d'instance. Lorsqu'une fenêtre d'événement est balisée avec une valeur spécifique, Amazon ECS mappe ces balises aux instances gérées Amazon ECS correspondantes des clusters correspondants et planifie la maintenance des instances pendant les périodes définies, dans la mesure du possible.

Pour plus d'informations sur les fenêtres d'événements, voir Créez des fenêtres d'événements personnalisées pour les événements planifiés qui concernent vos instances Amazon EC2 dans le Guide de l'utilisateur d'Amazon EC2.

Si des fenêtres d'événements existent, AWS ajuste le calendrier de vidange pour l'aligner sur ces fenêtres, ce qui peut entraîner un drainage plus tôt que la période de 14 jours par défaut pour respecter la fenêtre d'événements spécifiée. Les modifications de la fenêtre d'événements n'affectent que les instances de conteneurs gérés récemment lancées, garantissant ainsi une planification de maintenance prédictive.

Jusqu'à l'heure de drainage prévue, Amazon ECS poursuit les opérations normales de placement des tâches sur les instances de conteneur gérées conformément à la configuration du client.

Séquence de maintenance

À l'heure de maintenance désignée, Amazon ECS commence sa séquence de maintenance en invoquant l'API `UpdateContainerInstancesState` pour initier un drainage progressif de la charge de travail. Pendant la période de résiliation progressive, Amazon ECS tente d'arrêter la charge de travail sur l'instance marquée pour le drainage.

Amazon ECS arrête les tâches et lance les tâches de remplacement en fonction de la configuration de déploiement du service (`maximumPercent` et `minimumHealthyPercent`). Avec la valeur par défaut `maximumPercent` de 200 %, Amazon ECS planifie d'abord les nouvelles tâches et attend qu'elles soient exécutées avant d'arrêter les tâches existantes, minimisant ainsi les interruptions de service. Quand `maximumPercent` est réglé sur 100 %, Amazon ECS arrête d'abord les tâches, puis les remplace. Tout au long de ce processus, les services Amazon ECS respectent toutes les configurations de déploiement de service tout en poursuivant les tentatives de drainage jusqu'au 21e jour suivant le lancement de l'instance.

Si le drainage n'est pas terminé avant le 21e jour, Amazon ECS exécute l'API `DeregisterContainerInstance` pour arrêter l'instance de conteneur gérée et ses charges de travail restantes afin de maintenir la conformité et d'appliquer les derniers correctifs logiciels à l'instance gérée.

Considérations de sécurité pour les instances gérées Amazon ECS

Les instances gérées Amazon ECS fournissent une expérience de calcul de conteneur entièrement gérée qui vous permet d'exécuter des charges de travail sur des types d'instances Amazon EC2 spécifiques tout en confiant les responsabilités de sécurité à AWS. Cette rubrique décrit le modèle de sécurité, les fonctionnalités et les considérations à prendre en compte lors de l'utilisation des instances gérées Amazon ECS.

Pour plus d'informations sur AWS et les responsabilités des clients en matière de sécurité des instances gérées Amazon ECS, voir Modèle de responsabilité partagée pour les instances gérées Amazon ECS.

Modèle de sécurité

Les instances gérées Amazon ECS mettent en œuvre un modèle de sécurité complet qui concilie flexibilité et protection :

- AWS-infrastructure gérée - AWS contrôle le cycle de vie des instances gérées et gère les correctifs de sécurité, éliminant ainsi les risques d'erreur humaine et de falsification.
- Aucun accès administratif : le modèle de sécurité est verrouillé et interdit l'accès administratif aux instances gérées.
- Multi-task placement - Par défaut, Amazon ECS Managed Instances place plusieurs tâches sur une seule instance afin d'optimiser les coûts et l'utilisation, ce qui assouplit la contrainte d'isolation de la charge de travail par rapport à Fargate.
- Isolation des données - Bien que AWS contrôle le cycle de vie des instances et le placement des tâches, AWS Impossible de se connecter aux instances gérées ou d'accéder aux données des clients.

Comprendre les instances gérées

Amazon ECS Managed Instances fournit des instances gérées EC2 sur votre compte. En tant qu'opérateur désigné, Amazon ECS gère le cycle de vie complet de ces instances en votre nom, y compris le provisionnement, la mise à l'échelle, l'application de correctifs et la résiliation. Vous n'êtes pas autorisé à mettre fin directement à ces instances ou à modifier leurs paramètres. Pour plus d'informations, voir Instances gérées Amazon EC2.

Identification des instances gérées

Vous pouvez identifier les instances gérées Amazon ECS dans votre compte à l'aide des indicateurs suivants :

- Le `Operator` champ dans Amazon EC2 `DescribeInstances` réponse, avec une valeur de `ecs.amazonaws.com`.
- Le `aws:ec2:managed-launch` balise sur l'instance, avec une valeur de `ecs-managed-instances`.

Visibilité des ressources gérées

À compter du 22 avril 2026, Amazon EC2 masque les nouvelles instances gérées dans les vues de votre console Amazon EC2 et dans les opérations de liste d'API par défaut. Les paramètres de visibilité n'affectent pas la facturation ni le fonctionnement des ressources, et les instances gérées restent pleinement opérationnelles et facturables quelle que soit la configuration de visibilité. Vous pouvez modifier ce comportement à tout moment. Pour plus d'informations, voir Paramètres de visibilité des ressources gérées.

Fonctions de sécurité

Les instances gérées Amazon ECS incluent plusieurs fonctionnalités de sécurité intégrées conçues pour protéger vos charges de travail et maintenir un niveau de sécurité élevé. Ces fonctionnalités vont de l'application automatique de correctifs de sécurité à la prise en charge de fonctionnalités Linux privilégiées en cas de besoin.

Bonnes pratiques de sécurité

Les instances gérées sont configurées conformément à AWS les meilleures pratiques en matière de sécurité, notamment :

- Aucun accès SSH : l'accès au shell à distance est désactivé pour empêcher tout accès non autorisé.
- Système de fichiers racine immuable : le système de fichiers racine ne peut pas être modifié, ce qui garantit l'intégrité du système.
- Kernel-level contrôles d'accès obligatoires - SELinux fournit des mesures de sécurité supplémentaires au niveau du noyau.

Application de correctifs de sécurité automatiques

Les instances gérées Amazon ECS permettent d'améliorer le niveau de sécurité de vos charges de travail grâce à l'application automatique de correctifs :

- Mises à jour de sécurité régulières - Les instances sont régulièrement mises à jour avec les derniers correctifs de sécurité par AWS, en ce qui concerne les fenêtres de maintenance que vous configurez.

- Durée de vie limitée de l'instance : la durée de vie maximale d'une instance en cours d'exécution est limitée à 14 jours afin de garantir que les applications s'exécutent sur des instances correctement configurées et dotées des derniers correctifs de sécurité.
- Contrôle des fenêtres de maintenance : vous pouvez utiliser la fonctionnalité des fenêtres d'événements Amazon EC2 pour spécifier à quel moment Amazon ECS doit remplacer vos instances par des instances corrigées.

Fonctionnalités Linux privilégiées

Les instances gérées Amazon ECS prennent en charge les logiciels qui nécessitent des privilèges Linux élevés, permettant ainsi des solutions de surveillance et de sécurité avancées :

- Fonctionnalités prises en charge - Vous pouvez choisir d'accéder à toutes les fonctionnalités privilégiées de Linux, y compris `CAP_NET_ADMIN`, `CAP_SYS_ADMIN`, `CAP_BPF`, et `CAP_PERFMON`.
- Solutions populaires : cela vous permet d'exécuter des solutions populaires de surveillance et d'observabilité du réseau, telles que Wireshark et Datadog.
- Configuration explicite requise : vous devez configurer explicitement votre fournisseur de capacité d'instances gérées Amazon ECS pour activer les fonctionnalités Linux privilégiées, car cela peut présenter des risques de sécurité supplémentaires pour vos applications.

> **Important**
>
> L'activation de fonctionnalités Linux privilégiées peut exposer vos tâches à des risques de sécurité supplémentaires. N'activez ces fonctionnalités que lorsque vos applications l'exigent et assurez-vous de bien comprendre les implications en matière de sécurité.

Support en matière de conformité et de réglementation

Les instances gérées Amazon ECS maintiennent le même niveau de conformité qu'Amazon ECS :

- Programmes de conformité - Amazon ECS Managed Instances s'inscrit dans le même champ d'application AWS Programmes d'assurance tels qu'Amazon ECS, y compris PCI-DSS, HIPAA et FedRAMP.
- Points de terminaison FIPS - Les instances gérées Amazon ECS prennent en charge la configuration des terminaux FIPS au niveau du fournisseur de capacité. Contrairement à

Fargate, qui utilise un paramètre au niveau du compte, Amazon ECS Managed Instances utilise un paramètre par fournisseur de capacité, car FIPS est une configuration par instance. Vous configurez FIPS lors de la création ou de la mise à jour d'un fournisseur de capacité.

- Clés gérées par le client : les fonctionnalités de sécurité requises pour garantir la conformité, telles que les clés gérées par le client pour le chiffrement, sont prises en charge.

Instances gérées Amazon ECS FIPS-140 Considérations

Tenez compte des points suivants lorsque vous utilisez FIPS-140 conformité sur les instances gérées Amazon ECS :

- FIPS-140-compliant Les AMI des instances gérées sont disponibles dans AWS GovCloud (US) Régions uniquement.
- Les instances gérées Amazon ECS sont prises en charge FIPS-140-3
- FIPS-140 la conformité est activée par défaut dans AWS GovCloud (US) Régions. Si vous devez exécuter des charges de travail sans conformité FIPS, désactivez la conformité FIPS dans la configuration Managed Instances Capacity Provider.
- Le `cpuArchitecture` car vos tâches doivent être `X86_64` pour FIPS-140 conformité.

Désactiver FIPS sur les instances gérées Amazon ECS

Par défaut, les fournisseurs de capacité des instances gérées Amazon ECS dans AWS GovCloud (US) Lancement des régions FIPS-compliant AMI. Vous choisissez de désactiver FIPS-140 conformité lors de la création d'un nouveau fournisseur de capacités d'instances gérées Amazon ECS. Suivez ces étapes pour créer un nouveau fournisseur de capacité non conforme à la norme FIPS.

1. Désactiver FIPS-140 conformité sur Capacity Provider.

```
aws ecs create-capacity-provider \
    --cluster cluster-name \
    --name capacity-provider-name \
    --managed-instances-provider '{
        "infrastructureRoleArn": "infrastructure-role-arn",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "instance-profile-arn",
            "fipsEnabled": false,
```

```
            "networkConfiguration": {
                "subnets": ["subnet-id"],
                "securityGroups": ["security-group-id"]
            }
        }
    }'
```

2. Vous pouvez éventuellement utiliser ECS Exec pour exécuter la commande suivante afin de vérifier FIPS-140 statut de conformité d'un fournisseur de capacité.

   Remplacer *cluster-name* avec le nom de votre cluster, *task-id* avec l'ID ou l'ARN de votre tâche, et *container-name* avec le nom du conteneur de la tâche sur laquelle vous souhaitez exécuter la commande.

   Une valeur renvoyée de « 1 » indique que vous utilisez FIPS.

   ```
   aws ecs execute-command \
       --cluster cluster-name \
       --task task-id \
       --container container-name \
       --interactive \
       --command "cat /proc/sys/crypto/fips_enabled"
   ```

Considérations sur la sécurité

Lorsque vous utilisez des instances gérées Amazon ECS, vous devez comprendre et planifier plusieurs considérations de sécurité importantes. Ces considérations vous aident à prendre des décisions éclairées concernant l'architecture de votre charge de travail et les exigences de sécurité.

Multi-task modèle de sécurité

Le modèle de placement multitâche par défaut dans les instances gérées Amazon ECS est différent de l'isolation monotâche de Fargate :

- Ressources d'instance partagées : plusieurs tâches peuvent s'exécuter sur la même instance, ce qui peut exposer une tâche à des vulnérabilités émanant d'autres tâches exécutées sur la même instance ou dans le même cluster ECS.
- Single-task option - Vous pouvez configurer les instances gérées Amazon ECS pour utiliser le mode tâche unique pour les clients nécessitant le modèle de sécurité Fargate par défaut avec VM-level limite d'isolation de sécurité.

- Compromis entre coût et sécurité - Multi-task le mode permet d'optimiser les coûts et d'accélérer le démarrage des tâches, tandis que le mode tâche unique offre une meilleure isolation.

Gestion des interruptions d’instance

Il est important de concevoir vos applications de manière à tolérer les interruptions lorsque vous utilisez des instances gérées Amazon ECS :

- Tolérance aux interruptions : utilisez les instances gérées Amazon ECS avec des applications qui tolèrent l’interruption des services ou des tâches sous-jacents.
- Service-based charges de travail - Utilisez les services Amazon ECS pour le remplacement automatique des tâches, ou exécutez des charges de travail d'une durée contrôlée et limitée ne dépassant pas 14 jours sur des tâches autonomes.
- Arrêt progressif : configurez la période de grâce d’arrêt des tâches pour contrôler l’impact des interruptions.

Accès aux données et confidentialité

Les instances gérées Amazon ECS appliquent des contrôles d’accès aux données stricts :

- Aucun accès aux données des clients - Bien que AWS contrôle le cycle de vie des instances gérées et le placement des tâches sur les instances, AWS Impossible de se connecter aux instances gérées ou d'accéder aux données des clients.
- Métriques et journaux uniquement - AWS capture uniquement les métriques et les journaux associés nécessaires pour fournir les fonctionnalités d'Amazon ECS Managed Instances.
- Locked-down modèle de sécurité - Le modèle de sécurité interdit l'accès administratif, éliminant ainsi les risques d'erreur humaine et de falsification.

Bonnes pratiques de sécurité

Suivez ces pratiques exemplaires lors de l’utilisation d’instances gérées Amazon ECS :

- Évaluer le modèle de sécurité : prenez la décision consciente d’adopter les instances gérées Amazon ECS en fonction de vos exigences en matière de sécurité, notamment en ce qui concerne le modèle de placement multitâche.

- Utiliser le mode monotâche en cas de besoin : si vos charges de travail nécessitent une isolation renforcée, configurez les instances gérées Amazon ECS pour utiliser le mode monotâche.
- Minimiser les fonctionnalités privilégiées : n'activez les fonctionnalités Linux privilégiées que lorsque cela est absolument nécessaire et comprenez les risques de sécurité associés.
- Planifier les interruptions : concevez des applications capables de gérer correctement le remplacement des instances, en tenant particulièrement compte de la durée de vie maximale de 14 jours des instances.
- Configurer les fenêtres de maintenance : utilisez les fenêtres d'événements EC2 pour contrôler le moment où les remplacements d'instances ont lieu afin de minimiser l'impact sur vos charges de travail.
- Surveiller et auditer : passez régulièrement en revue la configuration de vos instances gérées Amazon ECS et surveillez tout événement ou modification lié à la sécurité.

Activation du contrôle du chiffrement VPC pour les instances gérées Amazon ECS

Les instances gérées Amazon ECS prennent en charge les contrôles de chiffrement VPC, une fonctionnalité de sécurité et de conformité qui fournit un contrôle centralisé pour surveiller et appliquer le chiffrement en transit pour tous les flux de trafic au sein et entre vos VPC d'une région. Lorsque VPC Encryption Controls est activé sur votre sous-réseau, vous pouvez spécifier des types d'instances prenant en charge le chiffrement en transit dans votre fournisseur de capacité personnalisé Amazon ECS Managed Instances, garantissant ainsi que les charges de travail Amazon ECS Managed Instances s'exécutent avec le chiffrement en transit.

Conditions préalables

Avant de commencer, vous avez besoin des éléments suivants :

- Un VPC dont le chiffrement en transit est activé sur les sous-réseaux. Pour plus d'informations, voir Documentation sur les contrôles de chiffrement VPC.
- Un fournisseur de capacité personnalisé Amazon ECS Managed Instances. Pour plus d'informations, consultez Architecture des instances gérées Amazon ECS.

Identifier les types d'instances compatibles

Les types d'instances Amazon EC2 doivent répondre à deux exigences :

1. Supporte le chiffrement VPC en transit - Utilisez ce qui suit AWS CLI commande pour répertorier les types d'instances Amazon EC2 qui prennent en charge le chiffrement en transit :

```
aws ec2 describe-instance-types \
    --filters Name=network-info.encryption-in-transit-supported,Values=true \
    --query "InstanceTypes[*].[InstanceType]" \
    --output text | sort
```

2. Pris en charge par les instances gérées Amazon ECS - Tous les types d'instances Amazon EC2 pris en charge par les instances gérées Amazon ECS sont documentés dans Types d'instances gérées Amazon ECS.

Si vous avez des exigences supplémentaires (telles que des besoins spécifiques en matière de processeur, de mémoire ou d'architecture), filtrez davantage les types d'instances compatibles en fonction de vos exigences en matière de charge de travail.

Création d'un cluster avec prise en charge du chiffrement VPC

Pour configurer les instances gérées Amazon ECS pour le chiffrement VPC en transit, procédez comme suit :

1. Créez un nouveau cluster et sélectionnez Fargate et instances gérées pour l'infrastructure.
2. Sélectionnez Utilisation personnalisée — avancée pour accéder à des paramètres de configuration supplémentaires.
3. Dans Types d'instances autorisés, ajoutez uniquement les types d'instances spécifiques qui prennent en charge le chiffrement VPC en transit.

Lorsqu'elles sont configurées de cette manière, les instances gérées Amazon ECS lancent uniquement les types d'instances Amazon EC2 qui prennent en charge le chiffrement VPC en transit.

Considérations

- Instances de performances explosibles - Les types d'instance T3, T3a et T4g ne prennent pas en charge le chiffrement VPC en transit et ne peuvent pas être utilisés dans des sous-réseaux où le contrôle du chiffrement est activé en mode imposé.
- Transitions de mode - Vous pouvez faire passer votre sous-réseau VPC du mode moniteur au mode imposé uniquement si toutes les instances en cours d'exécution prennent en charge le chiffrement VPC en transit.
- Échecs de lancement de tâches - En mode imposé, les tâches ne pourront pas être lancées si vous spécifiez des types d'instances qui ne prennent pas en charge le chiffrement en transit.

Résolution des problèmes

Échecs de lancement de tâches en mode imposé

Si les tâches ne sont pas lancées, vérifiez que tous les types d'instances spécifiés prennent en charge le chiffrement VPC en transit à l'aide du AWS CLI commande fournie ci-dessus.

Impossible de passer en mode forcé

Utilisez la console ou `GetVpcResourcesBlockingEncryptionEnforcement` commande permettant d'identifier les ressources qui n'appliquent pas le chiffrement en transit.

Pour plus d'informations sur les contrôles de chiffrement VPC, consultez le Documentation sur les contrôles de chiffrement VPC.

Optimisation de l’infrastructure des instances gérées Amazon ECS

Les instances gérées Amazon ECS provisionnent automatiquement des instances EC2 de taille appropriée en fonction de la configuration de votre fournisseur de capacité et des demandes de charge de travail actuelles, garantissant ainsi que vos applications conteneurisées disposent des ressources de calcul appropriées dès leur déploiement. Au fur et à mesure que les modèles de trafic de vos applications évoluent et que les exigences en matière de charge de travail évoluent au fil du temps, Amazon ECS Managed Instances surveille et optimise en permanence votre infrastructure en ajustant intelligemment la taille des instances pour répondre aux besoins actuels, en remplaçant de manière proactive les instances qui se sont éloignées des configurations optimales et en équilibrant dynamiquement la rentabilité, les performances des applications et la fiabilité du système. Ce

système de gestion des ressources fonctionne sans aucune intervention manuelle, ce qui réduit les coûts d'infrastructure tout en maintenant une haute disponibilité pour vos applications.

L'optimisation de l'infrastructure présente les avantages suivants :

- Optimisation des coûts : réduit les coûts d'infrastructure en maximisant l'utilisation des ressources et en éliminant les capacités inutilisées
- Amélioration des performances : optimise le placement de la charge de travail en fonction des besoins en ressources et des caractéristiques de performance
- Simplicité opérationnelle : automatise les décisions complexes de gestion des ressources sans intervention manuelle
- Amélioration de la fiabilité : maintient une haute disponibilité grâce à une répartition intelligente de la charge de travail et à une surveillance de l'état

Amazon ECS Managed Instances réalise deux types d'optimisations d'infrastructure afin de maximiser l'efficacité et de réduire les coûts :

Détection des instances inactives

Identifie et supprime les instances EC2 qui n'ont aucune tâche en cours d'exécution, éliminant ainsi les coûts d'infrastructure inutiles liés à la capacité inutilisée. Lorsqu'une instance inactive est détectée, le processus d'optimisation marque l'instance de conteneur comme DEREGISTERING, ce qui déclenche la séquence de nettoyage qui met fin en toute sécurité à l'instance EC2 sous-jacente.

Détection des instances sous-utilisées

Analyse la répartition des tâches entre les instances afin d'identifier les opportunités d'une meilleure allocation des ressources. Lorsque les tâches ne sont pas exécutées de manière optimale sur plusieurs instances, Amazon ECS Managed Instances consolide les charges de travail sur des instances moins nombreuses et utilisées plus efficacement, réduisant ainsi les coûts globaux tout en préservant les performances. Le processus d'optimisation marque les instances de conteneurs sous-utilisées comme étant DRAINING, ce qui déclenche le remplacement des tâches pour déplacer les charges de travail vers des instances existantes ou nouvelles, plus efficaces. Une fois que toutes les tâches ont été migrées en toute sécurité, l'instance passe à l'état DESREGISTERING et est nettoyée. Cette optimisation s'applique aux instances exécutant des tâches de service et garantit une consolidation sécurisée en respectant les limites de tâches minimale et maximale de votre service, en respectant le comportement de déploiement « démarrer-avant arrêt » et en conservant tous les

paramètres de protection des tâches tout au long du processus de vidange. Les instances exécutant des tâches autonomes ne sont pas prises en compte pour l'optimisation car les instances gérées par ECS ne remplacent pas les tâches autonomes.

> **Note**
>
> À compter du 26 octobre 2026, les instances gérées Amazon ECS consolident également les instances Spot sous-utilisées. Avant de vider une instance Spot, Amazon ECS Managed Instances lance la capacité de remplacement et attend que les tâches de remplacement soient opérationnelles. Ce n'est qu'alors qu'il arrête les tâches d'origine et récupère l'instance Spot. Cette approche du démarrage avant l'arrêt, associée aux limites de tâches minimales et maximales et aux paramètres de protection des tâches de votre service, permet de trouver un équilibre entre l'optimisation des coûts et la disponibilité des applications.

Ces optimisations fonctionnent ensemble pour garantir que votre infrastructure s'adapte en permanence aux demandes de charge de travail réelles, éliminant automatiquement le gaspillage et améliorant l'utilisation des ressources sans affecter la disponibilité des applications. Les deux mécanismes utilisent une surveillance pilotée par les événements qui répond aux événements du cycle de vie des tâches et des instances afin d'identifier les opportunités d'optimisation en temps réel. Amazon ECS Managed Instances détecte le moment où la dernière tâche s'arrête sur une instance de conteneur, ce qui indique une condition d'inactivité potentielle à des fins d'optimisation des coûts. Pour les instances sous-utilisées, tout arrêt de tâche ou tout lancement de nouvelle instance déclenche une analyse visant à identifier les opportunités de consolidation de la charge de travail et d'amélioration de l'efficacité des ressources.

ScaleInAfter

Les deux optimisations de l'infrastructure recherchent des opportunités de mettre fin aux instances en cours d'exécution afin d'améliorer l'utilisation et de réduire les coûts. Vous pouvez contrôler le calendrier de ces actions à l'aide du ScaleInAfter configuration dans les paramètres du fournisseur de capacité Amazon ECS Managed Instances, qui s'applique à la fois aux instances inactives et sous-utilisées. ScaleInAfter vous permet de spécifier le délai, en secondes, entre le moment où l'instance devient inactive ou sous-utilisée et le moment où Amazon ECS Managed Instances commence à optimiser votre infrastructure. Vous pouvez régler le délai entre 0 et 3600 secondes. Vous pouvez également spécifier -1 pour désactiver l'optimisation de l'infrastructure.

Instances inactives

- ECS attend la durée spécifiée après l'arrêt de la dernière tâche avant de désenregistrer l'instance
- Si une nouvelle tâche démarre pendant la période d'attente, l'instance n'est plus considérée comme inactive et la résiliation est annulée

Instances sous-utilisées

- ECS attend la durée spécifiée après un événement d'arrêt de tâche qui entraîne une sous-utilisation de l'instance avant de la vider
- Si une nouvelle tâche est lancée ou qu'une tâche existante est arrêtée sur une instance particulière pendant la période d'attente, le chronomètre est réinitialisé à partir de l'arrêt de la tâche le plus récent ou de l'heure de création de la nouvelle tâche, et Amazon ECS Managed Instances réévalue les inefficacités et prend les mesures nécessaires après l'expiration de la nouvelle période d'attente

Cette configuration est facultative. Lorsqu'elles ne sont pas spécifiées, les instances gérées ECS déterminent automatiquement la synchronisation optimale en fonction de la configuration par défaut des instances gérées ECS.

Réparation automatique du processeur graphique pour les instances gérées par Amazon ECS

Amazon ECS surveille l'état des GPU NVIDIA sur les instances gérées Amazon ECS qui utilisent du matériel GPU. Lorsqu'Amazon ECS détecte une défaillance matérielle du GPU, il peut automatiquement remplacer l'instance défectueuse. La réparation automatique du GPU est activée par défaut pour les instances gérées Amazon ECS.

Comment ça marche

Amazon ECS utilise NVIDIA Data Center GPU Manager (DCGM) pour surveiller l'état des GPU NVIDIA sur les instances gérées dotées d'un matériel GPU. Lorsque DCGM signale une défaillance critique du GPU, Amazon ECS marque l'instance comme étant altérée.

Amazon ECS surveille également les NVIDIA MPS démon sur les instances qui utilisent le partage GPU. Si le MPS le démon échoue, Amazon ECS marque l'instance comme étant altérée et la remplace en utilisant le même flux de travail.

Lorsque la réparation automatique du GPU est activée, Amazon ECS remplace l'instance altérée en utilisant un flux de travail « démarrer-before-stop » :

1. Amazon ECS définit l'instance altérée sur DRAINING. Aucune nouvelle tâche n'est placée sur l'instance.
2. Amazon ECS fournit une instance de remplacement.
3. Amazon ECS permet aux tâches existantes de s'arrêter correctement. Amazon ECS respecte le délai d'arrêt des tâches pour les tâches de l'instance.
4. Une fois la période de vidange terminée, Amazon ECS met fin à l'instance altérée.

Amazon ECS limite les actions de réparation afin d'empêcher les remplacements en cascade. Pas plus de 20 % des instances appartenant au fournisseur de capacité ne peuvent être vidées à la fois. S'il y a moins de 9 instances dans le fournisseur de capacité, au plus une instance est vidée à la fois.

Surveillance de l'état du GPU

Vous pouvez utiliser le `DescribeContainerInstances` API pour vérifier l'état du GPU. Pour plus d'informations, consultez Surveillance de l'état des instances de conteneurs Amazon ECS. Vous pouvez également surveiller l'évolution de l'état de santé du GPU via Événements de modification de l'état de santé des instances de conteneurs Amazon ECS.

Codes d'erreur XID surveillés

Amazon ECS surveille les codes d'erreur NVIDIA Xid suivants. Si Amazon ECS détecte l'une de ces erreurs, il marque l'instance comme étant défectueuse et la remplace.

Xid	Description
46	Le processeur graphique a arrêté le traitement
48	Erreur ECC à double bit
54	Connecteur d'alimentation auxiliaire non connecté
62	Arrêt du microcontrôleur interne
64	Échec de remappage de la mémoire GPU

Xid	Description
74	Erreur NVLink
79	Le GPU est tombé du bus
95	Erreur de mémoire non contenue
109	Délai d'expiration du changement de contexte
110	Le GPU a disparu du bus
136	La limite de retrait des pages de mémoire GPU est dépassée
140	Erreur ECC irrécupérable
142	Page de mémoire GPU supprimée en raison d'une erreur non corrigible
143	Page de mémoire GPU supprimée en raison d'un seuil d'erreur corrigible
151	Erreur d'interconnexion entre GPU et CPU
155	Erreur GPU NVLink flit CRC
156	Erreur de voie GPU NVLink
158	GPU InfoRoom endommagé

Pour plus d’informations sur les erreurs XID, consultez Erreurs XID dans la Documentation sur le déploiement et la gestion des GPU NVIDIA. Pour plus d’informations sur les messages XID individuels, consultez Comprendre les messages XID dans la Documentation sur le déploiement et la gestion des GPU NVIDIA.

Désactivation de la réparation automatique

La réparation automatique du GPU est activée par défaut pour les instances gérées Amazon ECS. Pour désactiver la réparation automatique du GPU, définissez `actionsStatus` pour `DISABLED`

dans `autoRepairConfiguration` lorsque vous créez ou mettez à jour un fournisseur de capacité. Vous pouvez également désactiver la réparation automatique du GPU dans la console Amazon ECS lorsque vous créez ou mettez à jour un fournisseur de capacité.

Lorsque la réparation automatique du GPU est désactivée, Amazon ECS continue de surveiller l'état de santé du GPU, mais ne remplace pas automatiquement les instances altérées.

> **Note**
>
> La désactivation de la réparation automatique du GPU désactive également la réparation automatique d'Amazon ECS Managed Daemons. Pour plus d'informations, voir Réparation automatique d'Amazon ECS Managed Daemons.

Pour désactiver la réparation automatique du GPU

```
aws ecs update-capacity-provider \
    --name my-gpu-capacity-provider \
    --managed-instances-provider '{
        "infrastructureRoleArn": "arn:aws:iam::111122223333:role/
ecsInfrastructureRole",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::111122223333:instance-profile/
ecsInstanceRole",
            "networkConfiguration": {
                "subnets": ["subnet-0123456789abcdef0"],
                "securityGroups": ["sg-0123456789abcdef0"]
            }
        },
        "autoRepairConfiguration": {
            "actionsStatus": "DISABLED"
        }
    }'
```

Pour activer la réparation automatique du GPU

```
aws ecs update-capacity-provider \
    --name my-gpu-capacity-provider \
    --managed-instances-provider '{
        "infrastructureRoleArn": "arn:aws:iam::111122223333:role/
ecsInfrastructureRole",
```

```
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::111122223333:instance-profile/
ecsInstanceRole",
            "networkConfiguration": {
                "subnets": ["subnet-0123456789abcdef0"],
                "securityGroups": ["sg-0123456789abcdef0"]
            }
        },
        "autoRepairConfiguration": {
            "actionsStatus": "ENABLED"
        }
    }'
```

Pour vérifier la configuration

```
aws ecs describe-capacity-providers \
    --capacity-providers my-gpu-capacity-provider
```

Migrer depuis AWS Fargate vers les instances gérées Amazon ECS

Vous pouvez procéder à la migration des charges de travail existantes de Fargate vers les instances gérées Amazon ECS. Cette migration vous permet d'accéder à la gamme complète des types d'instances Amazon EC2, des réservations de capacité et des fonctionnalités avancées tout en maintenant AWS-infrastructure gérée.

Considérations concernant la migration

Gardez les considérations suivantes à l'esprit lors de la migration de Fargate vers les instances gérées Amazon ECS :

Compatibilité de la tâche

> Les définitions de tâches existantes configurées pour Fargate sont pour la plupart compatibles avec les instances gérées Amazon ECS. Pour plus d'informations sur les différences entre les définitions de tâches, voir Différences entre les définitions de tâches Amazon ECS pour les instances gérées Amazon ECS.

Changements de modèle de sécurité

Les instances gérées Amazon ECS autorisent plusieurs tâches par instance par défaut. Envisagez d'activer le mode monotâche si vos charges de travail nécessitent une isolation renforcée.

Cycle de vie d'une instance

Les instances gérées Amazon ECS ont une durée de vie maximale de 14 jours. Planifiez le remplacement des tâches et utilisez les services Amazon ECS pour la gestion automatique des tâches.

Changements de tarification

Avec les instances gérées Amazon ECS, vous payez l'intégralité de l'instance plus des frais de gestion, plutôt que des ressources par tâche comme avec Fargate.

Fenêtres de maintenance

Configurez les fenêtres de maintenance à l'aide des fenêtres d'événements Amazon EC2 pour contrôler le moment où les instances gérées Amazon ECS sont remplacées pour l'application de correctifs.

Conditions préalables

Avant de procéder à la migration vers les instances gérées Amazon ECS, assurez-vous de disposer des éléments suivants :

- Les tâches Fargate existantes s'exécutent sur la version de plateforme 1.4.0 ou ultérieure
- Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :
  - Rôle de l'infrastructure - Permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

    Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
  - Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

    Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.
- Compréhension des différences de modèle de sécurité entre Fargate et les instances gérées Amazon ECS

> **Important**
>
> Les instances gérées Amazon ECS utilisent un modèle de sécurité différent de celui de Fargate. Les tâches multiples peuvent s'exécuter sur la même instance, ce qui peut exposer les tâches aux vulnérabilités émanant d'autres tâches. Passez en revue vos exigences en matière de sécurité avant de procéder à la migration.

Étape 1 : mettre à jour le cluster pour utiliser les instances gérées Amazon ECS

Créez un fournisseur de capacité. Lorsque vous créez un fournisseur de capacité avec les instances gérées Amazon ECS, celui-ci n'est disponible qu'au sein du cluster spécifié.

Pour plus d'informations, consultez Création d'un fournisseur de capacité pour les instances gérées Amazon ECS.

Étape 2 : mettre à jour la définition de tâche pour bénéficier de la fonctionnalité d'instances gérées Amazon ECS

Mettez à jour la définition de la tâche afin qu'elle possède les capacités requises pour les instances gérées Amazon ECS.

Pour plus d'informations, consultez Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console.

Étape 3 : mettre à jour le service pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS

Mettez à jour votre service Amazon ECS existant pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS.

Pour plus d'informations, consultez Mise à jour d'un service Amazon ECS pour utiliser un fournisseur de capacité.

Étape 4 : procéder à la migration des tâches autonomes

Pour les tâches autonomes, spécifiez le fournisseur de capacité des instances gérées Amazon ECS lors de l'exécution de la tâche.

Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.

Migration d'Amazon EC2 vers les instances gérées Amazon ECS

Procédez à la migration des charges de travail existantes depuis Amazon EC2 vers les instances gérées Amazon ECS. Cette migration vous permet d'accéder à la gamme complète des types d'instances Amazon EC2, des réservations de capacité et des fonctionnalités avancées tout en maintenant AWS-infrastructure gérée.

Considérations concernant la migration

Gardez les considérations suivantes à l'esprit lors de la migration d'Amazon EC2 vers des instances gérées Amazon ECS :

Compatibilité de la tâche

: Les définitions de tâches existantes configurées pour Amazon EC2 sont pour la plupart compatibles avec les instances gérées Amazon ECS. Pour obtenir la liste des différences entre les définitions de tâches, voir Différences entre les définitions de tâches Amazon ECS pour les instances gérées Amazon ECS.

Changements de modèle de sécurité

: Les instances gérées Amazon ECS autorisent plusieurs tâches par instance par défaut. Envisagez d'activer le mode monotâche si vos charges de travail nécessitent une isolation renforcée.

Cycle de vie d'une instance

: Les instances gérées Amazon ECS ont une durée de vie maximale de 14 jours. Planifiez le remplacement des tâches et utilisez les services Amazon ECS pour la gestion automatique des tâches.

Changements de tarification

: Avec Amazon ECS Managed Instances, vous payez l'intégralité de l'instance plus des frais de gestion, avec AWS gestion des frais généraux de gestion de l'infrastructure.

Fenêtres de maintenance

: Configurez les fenêtres de maintenance à l'aide des fenêtres d'événements Amazon EC2 pour contrôler le moment où les instances gérées Amazon ECS sont remplacées pour l'application de correctifs.

Conditions préalables

Avant de procéder à la migration vers les instances gérées Amazon ECS, assurez-vous de disposer des éléments suivants :

- Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :
  - Rôle de l'infrastructure - Permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

    Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
  - Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

    Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.
- Compréhension des différences de modèle de sécurité entre Amazon EC2 et les instances gérées Amazon ECS

Étape 1 : mettre à jour le cluster pour utiliser les instances gérées Amazon ECS

Créez un fournisseur de capacité. Lorsque vous créez un fournisseur de capacité avec les instances gérées Amazon ECS, celui-ci n'est disponible qu'au sein du cluster spécifié.

Pour plus d'informations, consultez Création d'un fournisseur de capacité pour les instances gérées Amazon ECS.

Étape 2 : mettre à jour la définition de tâche pour bénéficier de la fonctionnalité d'instances gérées Amazon ECS

Mettez à jour la définition de la tâche afin qu'elle possède les capacités requises pour les instances gérées Amazon ECS.

Pour plus d'informations, consultez Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console.

Étape 3 : mettre à jour le service pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS

Mettez à jour votre service Amazon ECS existant pour utiliser le fournisseur de capacité d'instances gérées Amazon ECS.

Pour plus d'informations, consultez Mise à jour d'un service Amazon ECS pour utiliser un fournisseur de capacité.

Étape 4 : procéder à la migration des tâches autonomes

Pour les tâches autonomes, spécifiez le fournisseur de capacité des instances gérées Amazon ECS lors de l'exécution de la tâche.

Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.

Démons gérés par Amazon ECS

Les Amazon ECS Managed Daemons vous permettent de déployer et de gérer des agents logiciels, tels que des agents de sécurité, d'observabilité et de mise en réseau, sur l'ensemble de votre infrastructure de conteneurs sur les instances gérées Amazon ECS. Les démons gérés dissocient la gestion du cycle de vie des démons des opérations applicatives. Vous pouvez déployer, mettre à jour et surveiller les agents de manière indépendante, sans redéployer les charges de travail ni coordonner les modifications entre les services.

Criticité des démons

Vous pouvez configurer chaque démon géré comme critique ou non critique, selon qu'une défaillance du démon doit entraîner la vidange de l'instance de conteneur par Amazon ECS.

- Critique : valeur par défaut. Amazon ECS démarre la tâche du démon avant de placer les tâches d'application sur l'instance, et si la tâche du démon s'arrête ou devient défectueuse, Amazon ECS vide et remplace l'instance de conteneur.
- Non-critical- Réglez le `critical` paramètre sur`false`. La tâche du démon fonctionne indépendamment de l'état de santé de l'instance du conteneur. Si une tâche de démon non critique échoue, s'arrête ou devient défectueuse, Amazon ECS maintient l'instance de conteneur active. Les tâches d'application existantes continuent de s'exécuter et Amazon ECS continue d'y placer de nouvelles tâches d'application. Un démon non critique ne bloque jamais l'enregistrement des instances, de sorte que les tâches de l'application peuvent être exécutées immédiatement même si le démon ne démarre pas.

Comment fonctionnent les démons gérés

Pour utiliser Managed Daemons, enregistrez d'abord une définition de tâche de démon. Une définition de tâche de démon est un modèle qui décrit les conteneurs qui forment un démon. Après avoir enregistré une définition de tâche de démon, créez un démon et associez-le à un cluster et à un ou plusieurs fournisseurs de capacité Amazon ECS Managed Instances. Amazon ECS garantit ensuite qu'une seule tâche de démon s'exécute sur chaque instance Amazon EC2 provisionnée par l'intermédiaire de ces fournisseurs de capacité.

Les démons ne lancent pas les instances de manière indépendante. Lorsque vous exécutez une tâche d'application sur un fournisseur de capacité Amazon ECS Managed Instances, Amazon

ECS provisionne une instance Amazon EC2, démarre la tâche du démon, puis transfère la tâche d'application vers. `RUNNING` Amazon ECS utilise cet ordre pour chaque démon, quelle que soit sa criticité, afin que les fonctions transversales telles que la journalisation, le suivi et la collecte de métriques soient opérationnelles avant que votre application ne commence à traiter les demandes.

La criticité contrôle ce que fait Amazon ECS lorsqu'une tâche de démon ne démarre pas ou s'arrête ultérieurement. Pour un démon critique, une tâche qui ne démarre pas empêche l'instance de devenir active. Une tâche qui s'arrête ou devient défectueuse une fois l'instance active entraîne la vidange et le remplacement de l'instance par Amazon ECS. Ce comportement de réparation automatique garantit une couverture fiable des démons sans intervention manuelle.

Un démon non critique ne bloque jamais l'instance et ne déclenche jamais de vidange. L'instance devient active même si la tâche du démon ne démarre pas et reste active si la tâche du démon s'arrête ou devient défectueuse ultérieurement. Les tâches de votre application continuent de s'exécuter et Amazon ECS continue de placer de nouvelles tâches d'application sur l'instance.

Lorsque vous mettez à jour un démon vers une nouvelle révision de définition de tâche, Amazon ECS effectue un déploiement continu sur toutes les instances des fournisseurs de capacité associés. Pendant le déploiement, Amazon ECS draine un pourcentage configurable d'instances simultanément, provisionne des instances de remplacement avec le démon mis à jour et remplace automatiquement vos tâches de service Amazon ECS. Amazon ECS fournit une protection intégrée contre les disjoncteurs. Vous pouvez configurer une durée de cuisson et CloudWatch des alarmes afin qu'Amazon ECS surveille le déploiement après la mise à jour de toutes les instances et annule automatiquement le déploiement en cas de problème. Amazon ECS émet un EventBridge événement et enregistre un journal des actions de service lorsqu'une tâche de démon ne démarre pas, pour les démons critiques et non critiques.

Principaux avantages

- Gestion du cycle de vie découplée : mettez à jour les démons indépendamment des déploiements d'applications.
- Couverture garantie : Amazon ECS lance la tâche du démon avant les tâches de l'application sur chaque instance, afin que les fonctions transversales soient opérationnelles avant que votre application ne commence à traiter les demandes. Pour un démon critique, une tâche de démon qui ne démarre pas empêche également l'instance de s'enregistrer, de sorte qu'une instance n'exécute jamais de tâches d'application sans son démon.

- Mises à jour de version fiables : lorsque vous mettez à jour la version d'un démon, Amazon ECS la déploie sur toutes les instances des fournisseurs de capacité associés, avec une protection intégrée contre les disjoncteurs et une restauration automatique pour garantir que chaque instance exécute la révision cible.
- Meilleure utilisation des ressources - L'exécution d'une seule tâche de démon par instance élimine le modèle sidecar par tâche, réduisant ainsi la charge de ressources sur l'ensemble de votre cluster.
- Réparation automatique des instances : pour un démon critique, si la tâche du démon s'arrête ou devient défectueuse, Amazon ECS vide et remplace automatiquement cette instance de conteneur. Cela permet de maintenir une couverture fiable des démons sans intervention manuelle.
- Continuité de la charge de travail : Non-critical les démons assurent le fonctionnement ininterrompu des charges de travail de vos applications, même si une tâche échoue, s'arrête ou devient défectueuse.

> **Note**
>
> Amazon ECS propose une stratégie de planification DAEMON pour les services Amazon ECS pour le type de lancement Amazon EC2. Managed Daemons est une nouvelle fonctionnalité conçue pour les instances gérées Amazon ECS afin de simplifier les déploiements de démons et de fournir de meilleures garanties de couverture des démons.

Prise en main

Pour commencer, enregistrez une définition de tâche de démon en spécifiant l'image de votre conteneur, puis créez un démon avec les fournisseurs de capacité associés dans votre cluster. Vous pouvez utiliser les kits de développement logiciel (SDK) Console de gestion AWS AWS CLI CloudFormation, ou les kits de AWS développement logiciel (SDK). Pour obtenir des instructions détaillées, consultez Création et gestion de démons.

Définitions de tâches de démon

La définition d'une tâche de démon constitue le plan directeur de votre démon. Les démons gérés Amazon ECS utilisent une ressource de définition de tâches de démon dédiée, distincte des définitions de tâches Amazon ECS standard. Vous enregistrez une définition de tâche de démon pour spécifier l'image du conteneur, les ressources requises et la configuration de votre démon.

Création d'une définition de tâche de démon

Vous pouvez créer une définition de tâche de démon en utilisant le Console de gestion AWS ou le AWS CLI.

Utilisation de Console de gestion AWS

1. Ouvrez la console Amazon ECS. Dans le volet de navigation, choisissez Définitions de tâches Daemon.
2. Choisissez Créer une nouvelle définition de tâche de démon.
3. Pour la famille de définitions de tâches Daemon, entrez un nom unique. Le nom peut contenir jusqu'à 255 caractères alphanumériques, traits d'union et traits de soulignement.
4. (Facultatif) Pour Rôle de tâche, choisissez un rôle IAM qui accorde des autorisations aux applications exécutées dans vos conteneurs. Laissez ce champ vide si vos conteneurs n'appellent pas d' AWS API.
5. (Facultatif) Pour Rôle d'exécution des tâches, sélectionnez`ecsTaskExecutionRole`. Amazon ECS requiert ce rôle pour extraire des images de conteneurs et publier des journaux.
6. (Facultatif) Dans la zone Taille de la tâche, spécifiez le processeur et la mémoire à réserver à votre démon. Par exemple, `0.25 vCPU` et `512 MB` la mémoire.
7. Configurez les détails de votre conteneur :
   - Nom du conteneur - Entrez un nom pour votre conteneur. Le nom peut contenir jusqu'à 255 caractères alphanumériques, traits d'union et traits de soulignement.
   - Contenant essentiel - Sélectionnez Oui. Chaque définition de tâche de démon nécessite au moins un conteneur essentiel. Si un conteneur essentiel tombe en panne, Amazon ECS arrête l'intégralité de la tâche.
   - URI de l'image : entrez l'URI de l'image Docker. Vous pouvez parcourir les images Amazon ECR ou utiliser des images publiques provenant de Docker Hub ou d'autres registres.
8. (Facultatif) Configurez l'allocation des ressources, le bilan de santé, les variables d'environnement, la collecte de journaux et les balises selon vos besoins.
9. Vérifiez votre configuration et choisissez Créer.

Utilisation de AWS CLI

Enregistrez une définition de tâche de démon en créant un fichier JSON et en utilisant la `register-daemon-task-definition` commande.

Voici un exemple de fichier JSON :

```
{
    "family": "my-daemon-task",
    "containerDefinitions": [
        {
            "name": "daemon-container",
            "image": "public.ecr.aws/docker/library/busybox:latest",
            "essential": true,
            "command": ["sh", "-c", "while true; do echo 'Daemon running'; sleep 30;
 done"],
            "memoryReservation": 512
        }
    ]
}
```

Exécutez la commande suivante pour enregistrer la définition de tâche du démon :

```
aws ecs register-daemon-task-definition --cli-input-json file://daemon-taskdef.json
```

Paramètres de définition des tâches du démon

Vous pouvez utiliser les paramètres suivants lorsque vous enregistrez une définition de tâche de démon.

Paramètres requis

- `family`- Le nom de famille de la définition de tâche. Cela regroupe plusieurs révisions de la même définition de tâche de démon.
- `containerDefinitions`- Un tableau d'`DaemonContainerDefinition`objets qui décrivent les conteneurs de votre tâche de démon.

Paramètres facultatifs

- `cpu`- Task-level Unités de traitement sous forme de chaîne (par exemple,`"256"`).
- `memory`- Task-level mémoire en MiB sous forme de chaîne (par exemple,`"512"`).
- `taskRoleArn`- L'ARN du rôle IAM qui accorde des autorisations aux conteneurs dans votre tâche de démon.

- `executionRoleArn`- L'ARN du rôle IAM que l'agent de conteneur Amazon ECS utilise pour effectuer des appels d' AWS API en votre nom (par exemple, extraire des images depuis Amazon ECR).
- `volumes`- Un éventail d'`DaemonVolume`objets. Les volumes Daemon ne prennent en charge que les montages de liaison avec `host` et`sourcePath`. Les définitions de tâches de démon ne prennent pas en charge Amazon EBS, Amazon EFS, FSx pour Windows File Server, les volumes Docker ou le stockage éphémère.
- `tags`- Un tableau de paires clé-valeur pour baliser la définition de la tâche de votre démon.
- `pidMode`- Contrôle le mode d'espace de noms PID pour le démon.
- `ipcMode`- Contrôle le mode d'espace de noms IPC pour le démon.

Mode réseau

Les démons utilisent un mode `daemon_bridge` réseau spécial qu'Amazon ECS définit automatiquement. Vous ne pouvez pas spécifier de mode réseau dans la définition de la tâche de votre démon. Tous les démons d'une instance partagent un espace de noms réseau unique et sont accessibles localement via une adresse IP de pont démon statique (`169.254.172.2`pour IPv4 ou pour IPv6). `fd00:ec2::172:2` Les tâches d'application (autres que les démons) peuvent communiquer avec les démons à cette adresse sans configuration réseau supplémentaire.

Étant donné que les démons s'exécutent dans un espace de noms réseau distinct de celui des tâches d'application, les tâches autres que les démons n'ont pas à se soucier des conflits de ports avec les démons. Cependant, étant donné que toutes les tâches du démon partagent le même espace de noms, les démons d'une même instance ne peuvent pas se lier au même port. Lorsque vous déployez plusieurs démons, assurez-vous que chaque démon utilise un port unique.

Capacités privilégiées

Sur les instances gérées Amazon ECS, les démons prennent en charge des fonctionnalités Linux privilégiées pour les opérations au niveau du système. Les agents de sécurité, les outils de surveillance du réseau et les agents d'observabilité nécessitent souvent un accès au niveau du noyau pour fonctionner correctement.

Vous pouvez octroyer des capacités de deux manières :

Le mode privilégié complet confère toutes les fonctionnalités Linux au conteneur. Utilisez-le lorsque votre agent a besoin d'un accès étendu au système :

```
{
    "containerDefinitions": [{
        "name": "security-daemon",
        "image": "my-security-agent:latest",
        "privileged": true
    }]
}
```

Les fonctionnalités individuelles n'accordent que les autorisations dont votre démon a besoin, selon le principe du moindre privilège. Utilisez le `linuxParameters.capabilities` champ pour ajouter des fonctionnalités individuelles telles que `SYS_ADMIN`NET_ADMIN,`SYS_PTRACE`,`BPF`, et `PERFMON` :

```
{
    "containerDefinitions": [{
        "name": "monitoring-daemon",
        "image": "my-monitoring-agent:latest",
        "linuxParameters": {
            "capabilities": {
                "add": ["NET_ADMIN", "SYS_PTRACE"]
            }
        }
    }]
}
```

Partage d'espaces de noms PID et IPC

Sur les instances gérées Amazon ECS, les démons prennent en charge le partage d'espaces de noms PID et IPC pour permettre des cas d'utilisation tels que les agents de sécurité, les outils de surveillance et les démons d'observabilité qui ont besoin de visibilité sur les processus de tâches des applications ou les ressources IPC.

`pidMode`

Type : chaîne

Obligatoire : non

Valeurs valides : `none` | `shared`

Par défaut: `none`

Mode d'espace de noms PID pour le démon. S'il `none` est spécifié ou si aucune valeur n'est fournie, le démon s'exécute avec son propre espace de noms PID, isolé des autres tâches. S'il `shared` est spécifié, le démon rejoint l'espace de noms PID de l'hôte, le rendant accessible aux tâches autres que les démons qui l'utilisent `pidMode: "host"` ou aux autres démons qui l'utilisent. `pidMode: "shared"`

`ipcMode`

Type : chaîne

Obligatoire : non

Valeurs valides : `none` | `shared`

Par défaut: `none`

Le mode d'espace de noms IPC pour le démon. S'il `none` est spécifié ou si aucune valeur n'est fournie, le démon s'exécute avec son propre espace de noms IPC, isolé des autres tâches. S'il `shared` est spécifié, le démon rejoint l'espace de noms IPC hôte, le rendant accessible aux tâches autres que des démons qui l'utilisent `ipcMode: "host"` ou aux autres démons qui l'utilisent. `ipcMode: "shared"`

L'exemple suivant montre une définition de tâche de démon avec le partage d'espaces de noms PID et IPC activé :

```
{
    "family": "my-monitoring-daemon",
    "pidMode": "shared",
    "ipcMode": "shared",
    "containerDefinitions": [
        {
            "name": "monitoring-agent",
            "image": "public.ecr.aws/my-company/monitoring-agent:latest",
            "essential": true
        }
    ]
}
```

Pour partager l'espace de noms PID avec un démon `pidMode` défini sur`shared`, une tâche qui n'est pas un démon doit le spécifier `pidMode: "host"` dans sa définition de tâche. Cela permet aux

conteneurs de la tâche de voir les processus du démon et permet des cas d'utilisation tels que la surveillance des processus et la livraison de signaux.

Pour partager l'espace de noms IPC avec un démon `ipcMode` défini sur`shared`, une tâche non démoniaque doit le spécifier `ipcMode: "host"` dans sa définition de tâche. Cela permet aux conteneurs de la tâche d'accéder aux ressources IPC du démon, telles que les segments de mémoire partagée et les files de messages.

Volumes

Les démons gérés prennent en charge les montages de liaisons à l'aide de volumes hôtes dotés d'une `sourcePath` spécification. Cela permet aux conteneurs de démons de monter des répertoires hôtes pour accéder aux journaux, aux métriques et aux informations système de l'instance Amazon EC2 sous-jacente. Les agents de collecte de journaux, les exportateurs de métriques et les scanners de sécurité l'utilisent généralement pour améliorer la visibilité des données au niveau de l'hôte.

La définition de tâche du démon prend en charge les spécifications du volume hôte avec`sourcePath`. Les volumes de démons persistent pendant tout le cycle de vie de l'instance de conteneur sous-jacente.

> **Note**
>
> Les volumes Amazon EBS, les volumes Amazon EFS, les volumes FSx pour Windows File Server, les volumes Docker et le stockage éphémère ne sont pas pris en charge dans les définitions de tâches de démon.

> **Note**
>
> Les démons ne prennent pas en charge le partage de volumes entre eux. Chaque démon fonctionne avec sa propre configuration de volume indépendante.

L'exemple suivant montre un démon Log Collector avec un montage bind :

```
{
    "containerDefinitions": [{
        "name": "log-collector",
        "image": "fluent/fluentd:latest",
```

```
        "mountPoints": [{
            "sourceVolume": "var-log",
            "containerPath": "/var/log"
        }]
    }],
    "volumes": [{
        "name": "var-log",
        "host": {
            "sourcePath": "/var/log"
        }
    }]
}
```

Paramètres de conteneur pris en charge

Les définitions de conteneurs de démons prennent en charge les paramètres suivants :

- `image`(obligatoire) - URI de l'image du conteneur
- `name`(obligatoire) - Nom du conteneur
- `cpu`- Unités CPU réservées au conteneur
- `memory`/`memoryReservation`- Limites de mémoire matérielle et logicielle
- `essential`- Si le contenant est essentiel (par défaut : vrai)
- `command`/`entryPoint`- Remplacer la commande ou le point d'entrée du conteneur
- `environment`/`environmentFiles`/`secrets`- Configuration de l'environnement
- `privileged`- Exécuter le conteneur en mode privilégié
- `user`- Utilisateur devant exécuter le conteneur en tant que
- `workingDirectory`- Répertoire de travail à l'intérieur du conteneur
- `readonlyRootFilesystem`- Montez le système de fichiers racine en lecture seule
- `mountPoints`- Points de montage du volume (liaison des montages avec host et SourcePath uniquement)
- `logConfiguration`- Configuration de la journalisation (supporte awslogs, Splunk, awsfirelens)
- `healthCheck`- Configuration du contrôle de santé des conteneurs
- `dependsOn`- Dépendances de démarrage des conteneurs
- `ulimits`- Limites de ressources
- `systemControls`- Paramètres du noyau

- `linuxParameters.capabilities`- Fonctionnalités Linux pour ajouter ou supprimer
- `linuxParameters.initProcessEnabled`- Exécuter un processus d'initialisation dans le conteneur
- `repositoryCredentials`- Informations d'identification pour les registres privés
- `restartPolicy`- Politique de redémarrage des conteneurs

Création et gestion de démons

Après avoir enregistré une définition de tâche de démon, créez un démon pour le déployer auprès de vos fournisseurs de capacité Amazon ECS Managed Instances. Amazon ECS place automatiquement une tâche de démon sur chaque instance Amazon EC2 des fournisseurs de capacité associés et gère le cycle de vie du démon.

Conditions préalables

Avant de créer un démon, vérifiez que vous disposez des éléments suivants :

- Un cluster Amazon ECS.
- Un ou plusieurs fournisseurs de capacité Amazon ECS Managed Instances associés au cluster.
- Une définition de tâche de démon enregistrée.

Création d'un démon

Console de gestion AWS

1. Ouvrez la console Amazon ECS. Dans le volet de navigation, choisissez Clusters, puis sélectionnez votre cluster.
2. Cliquez sur l'onglet Démons, puis sur Créer.
3. Pour la famille de définitions de tâches de démon, sélectionnez votre définition de tâche de démon dans la liste déroulante.
4. Pour la révision de la définition des tâches Daemon, sélectionnez la révision à utiliser. Laissez ce champ vide pour utiliser la dernière version.
5. Dans le champ Nom du démon, entrez un nom unique. Le nom peut contenir jusqu'à 255 caractères alphanumériques, traits d'union et traits de soulignement.

6. Pour les fournisseurs de capacité, sélectionnez un ou plusieurs fournisseurs de capacité Amazon ECS Managed Instances. Elles déterminent quelles instances exécutent les tâches de votre démon.
7. (Facultatif) Configurez les paramètres de déploiement :
   - Pourcentage de vidange : pourcentage d'instances à vider simultanément lors des mises à jour. Valeur par défaut : `25`.
   - Utiliser une ou plusieurs CloudWatch alarmes : activez cette option pour surveiller l'état du déploiement et revenez automatiquement en arrière si des alarmes se déclenchent.
   - Temps de cuisson : nombre de minutes pendant lesquelles Amazon ECS attend après avoir mis à jour toutes les instances avec la nouvelle révision du démon avant de terminer le déploiement. Pendant cette période, Amazon ECS surveille CloudWatch les alarmes et annule automatiquement le déploiement si une alarme se déclenche. Valeur par défaut : `0`.
8. (Facultatif) Dans Critical, choisissez si le démon est essentiel à l'état de santé de l'instance.
9. (Facultatif) Ajoutez des balises.
10. (Facultatif) Activez l'option Activer ECS Exec pour exécuter des commandes interactives dans vos conteneurs de démons à des fins de dépannage.
11. Vérifiez votre configuration et choisissez Créer.

AWS CLI

Créez un fichier JSON avec la configuration de votre démon et exécutez la `create-daemon` commande.

Voici un exemple de fichier JSON :

```
{
    "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/my-daemon-cluster",
    "daemonName": "my-monitoring-daemon",
    "daemonTaskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:daemon-task-
definition/my-daemon-task:1",
    "capacityProviderArns": [
        "arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-
provider"
    ],
    "critical": true
}
```

Exécutez la commande suivante pour créer le démon :

```
aws ecs create-daemon --cli-input-json file://create-daemon.json
```

Champs obligatoires

- `daemonName`- Un nom unique pour le démon.
- `clusterArn`- L'ARN du cluster.
- `daemonTaskDefinitionArn`- L'ARN de la définition de la tâche du démon.
- `capacityProviderArns`- Un ensemble d'ARN des fournisseurs de capacité Amazon ECS Managed Instances.

Champs facultatifs

- `deploymentConfiguration`- Un `DaemonDeploymentConfiguration` objet permettant de personnaliser le comportement de déploiement.
- `critical`- Si le démon est essentiel à la santé de l'instance.
- `tags`- Key-value paires pour le marquage.
- `propagateTags`- Paramètre de propagation des balises.
- `clientToken`- Un jeton d'idempotence.

Vérification du déploiement du démon

Après avoir créé un démon, vérifiez son statut en utilisant le Console de gestion AWS ou le AWS CLI.

Console de gestion AWS

1. Ouvrez la console Amazon ECS. Dans le volet de navigation, choisissez Clusters, puis sélectionnez votre cluster.
2. Choisissez l'onglet Démons.
3. Vérifiez que votre démon affiche le statut Actif.
4. Cliquez sur l'onglet Tâches pour confirmer qu'une tâche de démon est exécutée sur chaque instance de conteneur.

AWS CLI

Exécutez les commandes suivantes pour vérifier l'état du démon :

```
aws ecs list-daemons \
    --cluster-arn arn:aws:ecs:us-east-1:123456789012:cluster/my-daemon-cluster
```

```
aws ecs describe-daemons \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon
```

Mettre à jour un démon

Pour mettre à jour un démon avec une nouvelle révision de définition de tâche ou pour ajouter des fournisseurs de capacité, utilisez le Console de gestion AWS ou le AWS CLI. Cela déclenche un déploiement continu sur toutes les instances.

Console de gestion AWS

1. Ouvrez la console Amazon ECS. Dans le volet de navigation, choisissez Clusters, puis sélectionnez votre cluster.
2. Choisissez l'onglet Démons, puis sélectionnez le démon que vous souhaitez mettre à jour.
3. Choisissez Mettre à jour.
4. Pour la révision de la définition des tâches Daemon, sélectionnez la nouvelle révision.
5. (Facultatif) Mettez à jour les fournisseurs de capacité ou les paramètres de déploiement.
6. Choisissez Mettre à jour pour démarrer le déploiement continu.

AWS CLI

Exécutez la commande `update-daemon` :

```
aws ecs update-daemon \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon \
    --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-definition/my-daemon-task:2 \
```

```
    --capacity-provider-arns arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-provider \
    --no-critical
```

> **Important**
>
> Lorsque vous fournissez des paramètres de configuration de démon dans une `UpdateDaemon` demande, Amazon ECS utilise les paramètres que vous avez spécifiés au lieu des paramètres par défaut. Les paramètres de configuration du démon, y compris les balises et l'indicateur de commande enable execute, ne sont pas conservés entre les mises à jour. Chaque appel à `UpdateDaemon` doit inclure l'ensemble complet des paramètres de configuration que vous souhaitez appliquer. Tous les paramètres omis dans la demande sont rétablis à leurs valeurs par défaut. Le `critical` paramètre est défini par défaut sur`true`, donc si vous l'omettez dans une `UpdateDaemon` requête, le démon devient critique.

Supprimer un démon

Pour supprimer un démon, utilisez le Console de gestion AWS ou le AWS CLI. Attendez que toutes les tâches du démon s'arrêtent avant de supprimer le fournisseur de capacité ou le cluster.

Console de gestion AWS

1. Ouvrez la console Amazon ECS. Dans le volet de navigation, choisissez Clusters, puis sélectionnez votre cluster.
2. Choisissez l'onglet Démons, puis sélectionnez le démon que vous souhaitez supprimer.
3. Sélectionnez Delete (Supprimer).
4. Dans la boîte de dialogue de confirmation, choisissez Supprimer pour confirmer.

AWS CLI

Exécutez la commande `delete-daemon` :

```
aws ecs delete-daemon \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon
```

Déploiements de démons

Lorsque vous créez ou mettez à jour un démon, Amazon ECS effectue un déploiement continu pour exécuter des tâches de démon sur toutes les instances des fournisseurs de capacité associés. Cette section décrit le fonctionnement des déploiements de démons et explique comment configurer le comportement de déploiement.

Comment fonctionnent les déploiements de démons

Le déploiement d'un démon suit un modèle drain-approvisionnement-remplacement. Lorsque vous créez un démon sur un fournisseur de capacité avec des instances existantes, ou que vous mettez à jour un démon en fonction d'une nouvelle révision de définition de tâche, Amazon ECS vide un lot d'instances, provisionne des instances de remplacement avec le démon mis à jour et remplace automatiquement les tâches d'application. Ce processus se répète jusqu'à ce que toutes les instances exécutent la nouvelle révision du démon.

Le cycle de vie du déploiement passe par les états suivants :

1. `PENDING`- Amazon ECS a créé le déploiement et s'apprête à commencer.
2. `IN_PROGRESS`- Amazon ECS vide activement les instances et met en place des solutions de remplacement.
3. `SUCCESSFUL`- Toutes les instances exécutent la révision du démon cible.
4. `STOPPED`- Amazon ECS a arrêté le déploiement car il a échoué ou a été remplacé par un nouveau déploiement. Le démon peut être dans un état mixte où certaines instances exécutent l'ancienne révision et d'autres exécutent la nouvelle révision.

Amazon ECS revient automatiquement à la révision précédente du démon si le disjoncteur de déploiement détecte des défaillances ou si une CloudWatch alarme se déclenche pendant le déploiement.

Lors d'un déploiement, l'échec d'une tâche de démon non critique ne vide ni ne remplace l'instance. L'instance reste donc active et continue d'exécuter les tâches de votre application. Les échecs de lancement des tâches du démon sont toujours pris en compte pour le disjoncteur de déploiement, qui peut annuler une révision de cible instable.

Paramètres de configuration du déploiement

Vous pouvez personnaliser le comportement de déploiement en utilisant le `deploymentConfiguration` paramètre lorsque vous créez ou mettez à jour un démon.

- `drainPercent`(1.0—100,0) : pourcentage d'instances à vider simultanément pendant le déploiement. Des valeurs plus élevées accélèrent les déploiements mais peuvent réduire temporairement la capacité disponible. Par exemple, une valeur de `20.0` draine 20 % des instances à la fois. Si aucune valeur n'est spécifiée, la valeur par défaut est `25.0`.
- `alarms`(`DaemonAlarmConfiguration`) - CloudWatch alarmes à surveiller pendant le déploiement. Amazon ECS évalue les alarmes spécifiées pendant le déploiement et revient automatiquement en arrière si une alarme entre dans `ALARM` cet état. Amazon ECS ignore les alarmes qui étaient déjà en cours `ALARM` lorsque le déploiement commence.
- `bakeTimeInMinutes`(0—1440) - Nombre de minutes pendant lesquelles Amazon ECS attend après avoir mis à jour toutes les instances avec la nouvelle révision du démon avant de terminer le déploiement. Pendant cette période, Amazon ECS surveille CloudWatch les alarmes et annule automatiquement le déploiement si une alarme se déclenche. Si aucune valeur n'est spécifiée, la valeur par défaut est `0`.

Exemple avec configuration de déploiement :

```
aws ecs create-daemon \
    --cluster-arn arn:aws:ecs:us-east-1:123456789012:cluster/my-daemon-cluster \
    --daemon-name my-monitoring-daemon \
    --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-
definition/my-daemon-task:1 \
    --capacity-provider-arns arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-
daemon-capacity-provider \
    --deployment-configuration '{"drainPercent":20.0,"bakeTimeInMinutes":5}'
```

Surveillance des déploiements

Suivez la progression du déploiement à l'aide des commandes suivantes :

```
aws ecs list-daemon-deployments \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-
monitoring-daemon
```

```
aws ecs describe-daemon-deployments \
    --daemon-deployment-arns arn:aws:ecs:us-east-1:123456789012:daemon-deployment/my-daemon-cluster/abc123
```

La `describe-daemon-deployments` réponse inclut l'état du déploiement, la révision du démon cible et le `totalRunningInstanceCount` champ qui indique combien d'instances exécutent le démon cible.

Scénarios de déploiement de démons

Cette section décrit les scénarios de déploiement courants pour Managed Daemons.

- Déployer un démon sur un fournisseur de capacité vide
- Déployer un démon sur un fournisseur de capacité avec des instances existantes
- Ajouter un fournisseur de capacité à un démon existant
- Mettre à jour un démon vers une nouvelle révision

Déployer un démon sur un fournisseur de capacité vide

Dans ce scénario, vous déployez un démon sur un fournisseur de capacité sans instance existante. Les instances sont lancées lorsque vous planifiez des tâches d'application.

Conditions préalables : un cluster et un fournisseur de capacité Amazon ECS Managed Instances sans aucune instance en cours d'exécution.

1. Enregistrez la définition de la tâche de votre démon.
2. Créez le démon. Amazon ECS termine la création immédiatement même s'il n'existe pas encore d'instance.

```
aws ecs create-daemon \
    --cluster-arn arn:aws:ecs:us-east-1:123456789012:cluster/my-daemon-cluster \
    --daemon-name my-monitoring-daemon \
    --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-definition/my-daemon-task:1 \
    --capacity-provider-arns arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-provider
```

3. Créez un service qui utilise le même fournisseur de capacité. Lorsqu'Amazon ECS provisionne une instance pour le service, il démarre d'abord la tâche du démon, puis fait passer la tâche de l'application à`RUNNING`.

```
aws ecs create-service \
    --cluster my-daemon-cluster \
    --service-name my-app-service \
    --task-definition my-app-task \
    --desired-count 2 \
    --capacity-provider-strategy capacityProvider=my-daemon-capacity-
provider,weight=1
```

4. Vérifiez que la tâche du démon est exécutée sur chaque instance à l'aide de la `describe-daemons` commande ou en consultant l'onglet Démons de la console.

> **Note**
>
> Le déploiement du démon peut rester actif brièvement pendant qu'Amazon ECS provisionne des instances et démarre les tâches du démon. `PENDING` Attendez que le déploiement soit terminé `SUCCESSFUL` avant de vérifier le placement des tâches du démon.

Déployer un démon sur un fournisseur de capacité avec des instances existantes

Dans ce scénario, vous déployez un démon sur un fournisseur de capacité qui possède déjà des instances et des tâches d'application en cours d'exécution.

Conditions préalables : un cluster doté d'un fournisseur de capacité Amazon ECS Managed Instances qui possède des instances en cours d'exécution.

1. Enregistrez la définition de la tâche de votre démon.
2. Créez le démon. Amazon ECS lance un déploiement continu pour placer des tâches de démon sur toutes les instances existantes.

```
aws ecs create-daemon \
    --cluster-arn arn:aws:ecs:us-east-1:123456789012:cluster/my-daemon-cluster \
    --daemon-name my-monitoring-daemon \
    --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-
definition/my-daemon-task:1 \
```

```
    --capacity-provider-arns arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-provider \
    --deployment-configuration '{"drainPercent":25.0,"bakeTimeInMinutes":3}'
```

Amazon ECS draine un lot d'instances existantes (sur la base de`drainPercent`), provisionne des instances de remplacement avec le démon et remplace les tâches de l'application. Ce processus se répète jusqu'à ce que toutes les instances exécutent le démon.

3. Surveillez la progression du déploiement :

```
aws ecs list-daemon-deployments \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon
```

Ajouter un fournisseur de capacité à un démon existant

Dans ce scénario, vous ajoutez un second fournisseur de capacité à un démon existant. Le démon se déploie automatiquement sur les instances du nouveau fournisseur de capacité.

Prérequis : un démon s'exécutant sur un premier fournisseur de capacité et un second fournisseur de capacité créé et associé au cluster.

1. Mettez à jour le démon pour inclure les deux fournisseurs de capacité :

```
aws ecs update-daemon \
    --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon \
    --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-definition/my-daemon-task:1 \
    --capacity-provider-arns \
        arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-provider \
        arn:aws:ecs:us-east-1:123456789012:capacity-provider/my-daemon-capacity-provider-2
```

2. Surveillez le déploiement et vérifiez que les tâches du démon s'exécutent sur les instances des deux fournisseurs de capacité.

Mettre à jour un démon vers une nouvelle révision

Dans ce scénario, vous mettez à jour un démon existant pour utiliser une nouvelle révision de définition de tâche.

1. Enregistrez une nouvelle révision de la définition de la tâche de votre démon avec l'image ou la configuration du conteneur mise à jour.
2. Mettez à jour le démon pour utiliser la nouvelle révision :

   ```
   aws ecs update-daemon \
       --daemon-arn arn:aws:ecs:us-east-1:123456789012:daemon/my-daemon-cluster/my-monitoring-daemon \
       --daemon-task-definition-arn arn:aws:ecs:us-east-1:123456789012:daemon-task-definition/my-daemon-task:2
   ```

   Amazon ECS effectue un déploiement continu. Il vide les instances exécutant l'ancienne révision, provisionne les instances de remplacement avec la nouvelle révision et remplace automatiquement les tâches de l'application. Si le disjoncteur détecte des défaillances, Amazon ECS revient à la version précédente.
3. Surveillez le déploiement et vérifiez que toutes les instances exécutent la nouvelle révision :

   ```
   aws ecs describe-daemon-deployments \
       --daemon-deployment-arns arn:aws:ecs:us-east-1:123456789012:daemon-deployment/my-daemon-cluster/deployment-id
   ```

Événements Daemon

Amazon ECS envoie des événements de démon à afin de fournir une visibilité sur la gestion des tâches des démons et les modifications du cycle de vie du déploiement. Vous pouvez utiliser ces événements pour surveiller l'état des démons, suivre la progression du déploiement et déclencher des flux de travail automatisés.

Amazon ECS prend en charge les types d'événements de démon suivants :

- Événements d'action du service Daemon- Événements liés au placement des tâches du démon et aux problèmes de démarrage.
- Événements de changement d'état de déploiement du démon- Événements liés aux transitions du cycle de vie du déploiement des démons.

Pour savoir comment créer des règles pour ces événements, consultezCréation de règles pour les événements du démon.

Événements d'action du service Daemon

Amazon ECS envoie des événements d'action de service de démon avec le type `ECS Daemon Service Action` de détail. Ces événements vous avertissent lorsqu'Amazon ECS rencontre des problèmes lors du démarrage de tâches de démon sur vos instances de conteneur. Amazon ECS émet un EventBridge événement lorsqu'une tâche de démon ne démarre pas, à la fois pour les démons critiques et non critiques.

DAEMON_TASK_START_IMPAIRED

Amazon ECS envoie l'`DAEMON_TASK_START_IMPAIRED`événement lorsqu'il ne parvient pas à démarrer correctement une tâche de démon sur une instance de conteneur. L'événement inclut un `failureType` champ qui indique la cause de la panne :

- `TASK_FAILED_TO_RUN`- La tâche du démon a été créée mais n'a pas atteint `RUNNING` son statut. Les causes courantes incluent les échecs d'extraction de l'image du conteneur, les échecs de vérification de l'état des conteneurs ou les sorties essentielles des conteneurs. Le `taskArn` terrain est présent lors de l'événement.
- `TASK_CREATION_FAILED`- La tâche du démon n'a pas pu être créée sur l'instance du conteneur. Les causes courantes incluent l'insuffisance du processeur, de la mémoire ou d'autres ressources sur l'instance. Le `taskArn` champ n'est pas présent dans l'événement car aucune tâche n'a été créée.

Exemple : événement TASK_FAILED_TO_RUN

L'événement suivant montre une tâche démon qui a été créée mais qui n'a pas pu atteindre `RUNNING` son statut en raison d'un échec d'extraction de l'image du conteneur.

```
{
    "version": "0",
    "id": "12345678-1234-1234-1234-123456789012",
    "detail-type": "ECS Daemon Service Action",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-03-24T12:00:00Z",
```

```
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:123456789012:task/my-cluster/a1b2c3d4e5f6"
    ],
    "detail": {
        "eventType": "WARNING",
        "eventName": "DAEMON_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/my-cluster/a1b2c3d4e5f6",
        "taskArn": "arn:aws:ecs:us-west-2:123456789012:task/my-cluster/a1b2c3d4e5f6",
        "daemonRevisionArn": "arn:aws:ecs:us-west-2:123456789012:daemon-revision/my-
cluster/my-daemon/a1b2c3d4-e5f6-7890-abcd-ef1234567890",
        "capacityProviderName": "my-capacity-provider",
        "daemonArn": "arn:aws:ecs:us-west-2:123456789012:daemon/my-cluster/my-daemon",
        "daemonTaskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:daemon-task-
definition/my-daemon-td:1",
        "failureType": "TASK_FAILED_TO_RUN",
        "createdAt": "2026-03-24T12:00:00.000Z",
        "reason": "Task failed to reach RUNNING status: CannotPullContainerError: pull
 image manifest has been retried 5 time(s)"
    }
}
```


Exemple : événement TASK_CREATION_FAILED

L'événement suivant montre une tâche démon qui n'a pas pu être créée sur l'instance de conteneur en raison de ressources CPU insuffisantes. Le `taskArn` champ n'est pas présent car aucune tâche n'a été créée.


```
{
    "version": "0",
    "id": "87654321-4321-4321-4321-210987654321",
    "detail-type": "ECS Daemon Service Action",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-03-24T12:01:00Z",
    "region": "us-west-2",
    "resources": [],
    "detail": {
        "eventType": "WARNING",
        "eventName": "DAEMON_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
```

```
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-
instance/my-cluster/b2c3d4e5f6a7",
        "daemonRevisionArn": "arn:aws:ecs:us-west-2:123456789012:daemon-revision/my-
cluster/my-daemon/a1b2c3d4-e5f6-7890-abcd-ef1234567890",
        "capacityProviderName": "my-capacity-provider",
        "daemonArn": "arn:aws:ecs:us-west-2:123456789012:daemon/my-cluster/my-daemon",
        "daemonTaskDefinitionArn": "arn:aws:ecs:us-west-2:123456789012:daemon-task-
definition/my-daemon-td:1",
        "failureType": "TASK_CREATION_FAILED",
        "createdAt": "2026-03-24T12:01:00.000Z",
        "reason": "RESOURCE:CPU - Unable to place daemon task on container instance:
 insufficient CPU"
    }
}
```

Événements de changement d'état de déploiement du démon

Amazon ECS envoie des événements de changement d'état de déploiement du démon avec le type `ECS Daemon Deployment State Change` de détail. Amazon ECS émet ces événements chaque fois que l'état du déploiement d'un démon change.

Types d'événements de changement d'état de déploiement

Amazon ECS classe les événements de changement d'état de déploiement selon les types d'événements suivants :

Évènements INFO

- `DAEMON_DEPLOYMENT_PENDING`- Amazon ECS a lancé le déploiement d'un démon.
- `DAEMON_DEPLOYMENT_IN_PROGRESS`- Amazon ECS a commencé le déploiement et remplace activement les tâches du démon.
- `DAEMON_DEPLOYMENT_SUCCESSFUL`- Amazon ECS a terminé le déploiement avec succès, toutes les tâches du démon étant en cours d'exécution et en bon état.
- `DAEMON_DEPLOYMENT_STOPPED`- Amazon ECS a arrêté le déploiement. Cela se produit lorsqu'un déploiement a échoué ou a été remplacé par un nouveau déploiement.
- `DAEMON_DEPLOYMENT_STOP_REQUESTED`- Amazon ECS a interrompu le déploiement et va maintenant commencer à revenir en arrière.
- `DAEMON_DEPLOYMENT_ROLLBACK_IN_PROGRESS`- Amazon ECS a lancé une restauration en raison d'un échec de déploiement ou d'un déclenchement par disjoncteur.

- `DAEMON_DEPLOYMENT_ROLLBACK_SUCCESSFUL`- Amazon ECS a terminé avec succès la restauration du déploiement.

Evénements ERROR

- `DAEMON_DEPLOYMENT_ROLLBACK_FAILED`- Amazon ECS n'a pas pu terminer la restauration du déploiement.

Exemple : événement en attente de déploiement

L'événement suivant montre le déploiement d'un démon initié par Amazon ECS.

```
{
    "version": "0",
    "id": "3329f79b-3dca-07f8-b1c2-5fe99f0b5e87",
    "detail-type": "ECS Daemon Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2026-03-05T15:54:41Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:daemon/my-cluster/my-daemon"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "DAEMON_DEPLOYMENT_PENDING",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/my-cluster",
        "daemonArn": "arn:aws:ecs:us-west-2:111122223333:daemon/my-cluster/my-daemon",
        "daemonDeploymentArn": "arn:aws:ecs:us-west-2:111122223333:daemon-
deployment/my-cluster/my-daemon/0EYSiB0qap8xf0N76FsbE",
        "targetDaemonRevisionArn": "arn:aws:ecs:us-west-2:111122223333:daemon-
revision/my-cluster/my-daemon/85707969-3732-4b6a-a37d-5cefddd7d7dd",
        "updatedAt": "2026-03-05T15:54:41.618059641Z"
    }
}
```

Création de règles pour les événements du démon

Vous pouvez créer des règles pour recevoir des notifications lorsque des événements de démon se produisent. Pour plus d'informations sur la création de règles, consultez la section Création d'une règle dans le Guide de EventBridge l'utilisateur Amazon.

Les exemples de modèles d'événements suivants montrent comment filtrer les événements des démons.

Exemple : faire correspondre tous les événements d'action du service du démon

```
{
    "source": ["aws.ecs"],
    "detail-type": ["ECS Daemon Service Action"]
}
```

Exemple : correspondre à un type de défaillance spécifique

```
{
    "source": ["aws.ecs"],
    "detail-type": ["ECS Daemon Service Action"],
    "detail": {
        "eventName": ["DAEMON_TASK_START_IMPAIRED"],
        "failureType": ["TASK_CREATION_FAILED"]
    }
}
```

Exemple : faire correspondre des événements pour un cluster spécifique

```
{
    "source": ["aws.ecs"],
    "detail-type": ["ECS Daemon Service Action"],
    "detail": {
        "eventName": ["DAEMON_TASK_START_IMPAIRED"],
        "clusterArn": ["arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster"]
    }
}
```

Exemple : faire correspondre tous les événements de changement d'état du déploiement

```
{
    "source": ["aws.ecs"],
    "detail-type": ["ECS Daemon Deployment State Change"]
}
```

Métriques des démons

Amazon ECS publie des CloudWatch métriques pour Managed Daemons. Vous pouvez utiliser ces mesures pour suivre l'utilisation du processeur et de la mémoire pour les tâches des démons sur l'ensemble de votre cluster.

Les métriques Daemon utilisent le même `AWS/ECS` espace de noms que les métriques de service. Amazon ECS distingue les métriques du démon des métriques de service par la valeur de la `ServiceName` dimension, qui utilise le préfixe `daemon:` suivi du nom du démon (par exemple,). `daemon:`*`my-daemon`*

Métriques d'utilisation des démons

Amazon ECS mesure l'utilisation du processeur et de la mémoire des démons en tant que pourcentage de ressources utilisées par les tâches du démon par rapport aux ressources spécifiées dans la définition des tâches du démon.

CloudWatch fournit les statistiques suivantes pour les mesures d'utilisation des démons :

- Moyenne : utilisation moyenne de toutes les tâches du démon du cluster.
- Minimum : utilisation de la tâche du démon qui utilise le moins de ressources.
- Maximum : utilisation de la tâche du démon qui utilise le plus de ressources.

Formules d'utilisation

Amazon ECS calcule la statistique moyenne à l'aide des formules suivantes :

```
                                    (Total CPU units used by daemon tasks) x 100
Daemon CPU utilization =
 ---------------------------------------------------------------------------
```

```
                          (Total CPU units specified in daemon task definition) x
 (Number of daemon tasks)
```

```
                                        (Total MiB of memory used by daemon tasks) x
 100
Daemon memory utilization =
 ------------------------------------------------------------------------------
                            (Total MiB of memory specified in daemon task definition)
 x (Number of daemon tasks)
```

Ces formules s'appliquent uniquement à la statistique moyenne. Les statistiques Minimum et Maximum représentent la tâche de démon individuelle dont l'utilisation des ressources est la plus faible et la plus élevée, respectivement.

L'agent de conteneur Amazon ECS échantillonne l'utilisation du processeur et de la mémoire pour chaque tâche du démon en cours d'exécution toutes les 20 secondes. Chaque minute, l'agent transmet trois échantillons à Amazon ECS. Amazon ECS regroupe les échantillons de toutes les tâches de démon du cluster et publie les valeurs moyenne, minimale et maximale en pourcentage des ressources totales spécifiées dans la définition des tâches du démon. CloudWatch

Exemple : calcul de l'utilisation

Une définition de tâche de démon spécifie 256 unités centrales et 512 Mo de mémoire. Le démon a 2 tâches en cours d'exécution sur 2 instances de conteneur. Une tâche de démon utilise 200 unités centrales et 400 Mo de mémoire. L'autre utilise 100 unités centrales et 256 Mo de mémoire. Amazon ECS fournit les statistiques suivantes :

- Utilisation moyenne du processeur : ((200 + 100) x 100)/(256 x 2) = 58,6 %
- Utilisation moyenne de la mémoire : ((400 + 256) x 100)/(512 x 2) = 64,1 %

Affichage des métriques des démons

Vous pouvez consulter les métriques des démons en utilisant le Console de gestion AWS ou le AWS CLI.

Console de gestion AWS

1. Ouvrez la CloudWatch console.

2. Dans le volet de navigation, choisissez Metrics, puis All metrics.
3. Choisissez l'espace de nom AWS/ECS.
4. Choisissez ClusterName la ServiceName dimension.
5. Filtrez les `ServiceName` valeurs commençant par « `daemon:` Pour localiser les métriques de votre démon ».

AWS CLI

Exécutez la `get-metric-statistics` commande pour récupérer les métriques du CloudWatch démon. Spécifiez la `ServiceName` dimension à l'aide du `daemon:` préfixe.

Exemple : affichage de l'utilisation du processeur d'un démon

La commande suivante récupère l'utilisation moyenne du processeur pour un démon sur une période de 12 heures avec des intervalles de 5 minutes.

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/ECS \
    --metric-name CPUUtilization \
    --dimensions Name=ClusterName,Value=my-cluster Name=ServiceName,Value=daemon:my-
daemon \
    --start-time 2026-03-24T00:00:00Z \
    --end-time 2026-03-24T12:00:00Z \
    --period 300 \
    --statistics Average
```

Exemple : affichage de l'utilisation de la mémoire d'un démon

La commande suivante récupère l'utilisation moyenne de la mémoire par un démon sur une période de 12 heures avec des intervalles de 5 minutes.

```
aws cloudwatch get-metric-statistics \
    --namespace AWS/ECS \
    --metric-name MemoryUtilization \
    --dimensions Name=ClusterName,Value=my-cluster Name=ServiceName,Value=daemon:my-
daemon \
    --start-time 2026-03-24T00:00:00Z \
    --end-time 2026-03-24T12:00:00Z \
    --period 300 \
```

```
    --statistics Average
```

Réparation automatique Daemon

Un démon peut être critique ou non pour l'état de santé de l'instance, que vous contrôlez à l'aide du `critical` paramètre lorsque vous créez ou mettez à jour le démon. Le `critical` paramètre est défini par défaut sur`true`, donc un démon est essentiel, sauf si vous le définissez sur`critical`. `false`

Pour un démon critique, si la tâche du démon s'arrête ou devient défectueuse, Amazon ECS considère que l'instance est altérée et la vide automatiquement et la remplace. Les actions de réparation automatique d'un démon critique sont les suivantes :

1. Amazon ECS détecte lorsqu'une tâche de démon s'arrête ou devient défectueuse.
2. Amazon ECS marque l'instance comme étant épuisante, ce qui l'empêche d'accepter de nouvelles tâches d'application.
3. Amazon ECS provisionne une instance de remplacement et lance la tâche du démon sur celle-ci.
4. Une fois que la tâche du démon a atteint un état normal, Amazon ECS planifie les tâches de l'application depuis l'instance de vidange vers l'instance de remplacement.
5. Amazon ECS met fin à l'instance d'origine.

Pour un démon non critique, la tâche du démon fonctionne indépendamment de l'état de santé de l'instance du conteneur. Si la tâche du démon échoue, s'arrête ou devient défectueuse, Amazon ECS maintient l'instance de conteneur active. Un démon non critique ne bloque jamais l'enregistrement des instances, de sorte que les tâches de l'application peuvent être exécutées immédiatement même si le démon ne démarre pas.

> **Important**
>
> Les vérifications de l'état des démons sont facultatives mais vivement recommandées. Sans bilan de santé, Amazon ECS ne peut détecter les défaillances que lorsque la tâche du démon s'arrête.

Vous pouvez surveiller l'état des démons à l'aide de l'`DescribeContainerInstances`API ou de l'`DescribeTasks`API.

Pour un démon non critique, Amazon ECS maintient l'instance active lorsque la tâche du démon n'est pas en cours d'exécution. Les champs suivants indiquent le nombre d'instances se trouvant dans cet état :

- `DescribeDaemonwithoutDaemonCount`les rendements pour chaque fournisseur de capacité et `totalWithoutDaemonCount` pour la révision pour tous les fournisseurs de capacité.
- `DescribeDaemonDeploymentswithoutDaemonInstanceCount`les rendements pour chaque fournisseur de capacité et `totalWithoutDaemonInstanceCount` pour la révision pour tous les fournisseurs de capacité.

> **Note**
>
> Le nombre de démons sans démon est distinct du nombre en cours d'exécution. `DescribeDaemon`rapports `runningCount` et`totalRunningCount`. `DescribeDaemonDeployments`rapports `runningInstanceCount` et`totalRunningInstanceCount`. Une instance apparaît dans un ensemble ou dans l'autre, jamais dans les deux. N'additionnez donc pas les deux ensembles.

Architecte pour AWS Fargate pour Amazon ECS

AWS Fargate est une technologie que vous pouvez utiliser avec Amazon ECS pour exécuter des conteneurs sans avoir à gérer des serveurs ou des clusters d'instances Amazon EC2. Avec AWS Fargate, vous n'avez plus besoin d'allouer, de configurer ou de mettre à l'échelle des clusters de machines virtuelles pour exécuter des conteneurs. Vous n'avez plus à choisir de types de serveurs, décider quand mettre à l'échelle vos clusters ni optimiser les packs de clusters.

Lorsque vous exécutez vos tâches et services avec Fargate, vous packagez votre application dans des conteneurs, spécifiez les exigences en matière d'UC et de mémoire, définissez les politiques de mise en réseau et les politiques IAM, puis lancez l'application. Chaque tâche Fargate possède sa propre limite d'isolement et ne partage pas le noyau sous-jacent, les ressources de processeur, les ressources de mémoire ou l'interface réseau Elastic avec une autre tâche. Vous configurez vos définitions de tâches pour Fargate en définissant le paramètre de définition de tâche `requiresCompatibilities` sur `FARGATE`. Pour plus d'informations, consultez Capacity.

Fargate propose des versions de plateforme pour Amazon Linux 2 (version de plateforme 1.3.0), le système d'exploitation Bottlerocket (version de plateforme 1.4.0) et Microsoft Windows 2019 Server Full et Core Editions. Sauf indication contraire, les informations s'appliquent à toutes les plateformes Fargate.

Pour de plus amples informations sur les Régions qui prennent en charge les conteneurs Linux sur Fargate, veuillez consulter the section called "Conteneurs Linux sur AWS Fargate".

Pour de plus amples informations sur les Régions qui prennent en charge les conteneurs Windows sur Fargate, veuillez consulter the section called "Conteneurs Windows allumés AWS Fargate".

Procédures

Pour plus d'informations sur la manière de démarrer avec la console, consultez la section :

- Création d'une tâche Amazon ECS Linux pour Fargate
- Création d'une tâche Windows Amazon ECS pour Fargate

Pour plus d'informations sur la façon de commencer à utiliser AWS CLI, voir :

- Création d'une tâche Linux Amazon ECS pour le Fargate avec AWS CLI
- Création d'une tâche Windows Amazon ECS pour le Fargate avec AWS CLI

Fournisseurs de capacité

Les fournisseurs de capacité suivants sont disponibles :

- Fargate
- Fargate Spot : exécutez des tâches Amazon ECS tolérantes aux interruptions à un tarif réduit par rapport au prix AWS Fargate. Fargate Spot exécute les tâches sur la capacité de calcul de réserve. Quand AWS Si la capacité est rétablie, vos tâches seront interrompues avec un avertissement de deux minutes. Pour plus d'informations, consultez Clusters Amazon ECS pour Fargate.

Définitions de tâche

Les tâches Fargate ne prennent pas en charge tous les paramètres de définition de tâche Amazon ECS disponibles. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment pour les tâches Fargate. Pour plus d'informations, consultez UC et mémoire au niveau de la tâche.

Versions de plateforme

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Il s'agit d'une combinaison de la version Kernel et de la version d'exécution du conteneur. Vous sélectionnez une version de plateforme lorsque vous exécutez une tâche ou lorsque vous créez un service pour gérer un certain nombre de tâches identiques.

De nouvelles révisions des versions de plateforme sont publiées au fil de l'évolution de l'environnement d'exécution, par exemple si des mises à jour, de nouvelles fonctionnalités, des corrections de bugs ou des mises à jour de sécurité sont apportées au noyau ou au système d'exploitation. Une version de plateforme Fargate est mise à jour en effectuant une nouvelle révision de la version de plateforme. Chaque tâche s'exécute sur une révision de version de plateforme au cours de son cycle de vie. Si vous souhaitez utiliser la dernière version de plateforme, vous devez démarrer une nouvelle tâche. Une nouvelle tâche exécutée sur Fargate s'exécute toujours sur la dernière version de plateforme, ce qui garantit que les tâches sont toujours lancées sur une infrastructure sécurisée et corrigée.

Si un problème de sécurité affectant une version existante de la plateforme est détecté, AWS crée une nouvelle révision corrigée de la version de la plateforme et supprime les tâches exécutées sur

la révision vulnérable. Dans certains cas, vous pouvez être averti que la résiliation de vos tâches sur Fargate a été planifiée. Pour plus d'informations, consultez Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.

Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.

Équilibrage de charge des services

Vous pouvez configurer votre service Amazon ECS service sur AWS Fargate de manière à utiliser Elastic Load Balancing pour répartir uniformément le trafic entre les tâches de votre service.

Les services Amazon ECS sur AWS Fargate prennent en charge les types d'équilibreurs de charge Application Load Balancer, Network Load Balancer et Gateway Load Balancer. Les équilibreurs de charge d'application sont utilisés pour acheminer HTTP/HTTPS trafic (ou couche 7). Les équilibreurs de charge Network Load Balancer permettent d'acheminer le trafic TCP ou UDP (ou couche 4). Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.

De même, lorsque vous créez des groupes cible pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.

L'utilisation d'un Network Load Balancer pour acheminer le trafic UDP vers votre Amazon ECS sur les tâches AWS Fargate n'est prise en charge que lors de l'utilisation de la plateforme version 1.4 ou ultérieure.

Métriques d'utilisation

Vous pouvez utiliser CloudWatch des statistiques d'utilisation pour fournir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces statistiques pour visualiser votre utilisation actuelle des services sur CloudWatch graphiques et tableaux de bord.

AWS Fargate les mesures d'utilisation correspondent à AWS quotas de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d'un quota de service. Pour plus d'informations sur les quotas de service AWS Fargate, consultez la section Points de terminaison et quotas Amazon ECS dans la Référence générale d'Amazon Web Services.

Pour plus d'informations sur AWS Fargate statistiques d'utilisation, voir AWS Fargate mesures d'utilisation.

Considérations relatives à la sécurité d'Amazon ECS concernant l'utilisation de Fargate

Nous recommandons aux clients qui recherchent une isolation forte pour leurs tâches d'utiliser Fargate. Fargate exécute chaque tâche dans un environnement de virtualisation matérielle. Cela garantit que ces charges de travail conteneurisées ne partagent pas les interfaces réseau, le stockage éphémère Fargate, l'UC ou la mémoire avec d'autres tâches.

Pratiques exemplaires en matière de sécurité Fargate dans Amazon ECS

Nous vous recommandons de prendre en compte les bonnes pratiques ci-dessous lorsque vous utilisez AWS Fargate.

Utilisation AWS KMS pour crypter le stockage éphémère pour Fargate

Votre stockage éphémère devrait être chiffré soit par AWS KMS ou vos propres clés gérées par le client. Pour les tâches hébergées sur Fargate à l'aide de la version de plateforme `1.4.0` ou ultérieure, chaque tâche bénéficie de 20 Gio de stockage éphémère. Pour plus d'informations, consultez la section Clé gérée par le client (CMK). Vous pouvez augmenter la quantité totale de stockage éphémère, jusqu'à un maximum de 200 Gio, en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche. Pour les tâches qui ont été lancées le 28 mai 2020 ou ultérieurement, le stockage éphémère est crypté à l'aide d'un AES-256 algorithme de chiffrement utilisant une clé de chiffrement gérée par Fargate.

Pour plus d'informations, consultez la section Options de stockage pour les tâches Amazon ECS.

Exemple : lancement d'une tâche sur la version de plateforme Fargate 1.4.0 avec chiffrement du stockage éphémère

La commande suivante lancera une tâche Amazon ECS sur la version de plateforme Fargate 1.4. Cette tâche étant lancée dans le cadre du cluster, elle utilise les 20 Gio de stockage éphémère qui sont automatiquement soumis au chiffrement.

```
aws ecs run-task --cluster clustername \
  --task-definition taskdefinition:version \
  --count 1
  --launch-type "FARGATE" \
  --platform-version 1.4.0 \
  --network-configuration
 "awsvpcConfiguration={subnets=[subnetid],securityGroups=[securitygroupid]}" \
  --region region
```

Fonctionnalité SYS_PTRACE pour le suivi des appels système du noyau avec Fargate

La configuration par défaut des fonctionnalités Linux ajoutées ou supprimées de votre conteneur est fournie par Docker.

Les tâches lancées sur Fargate ne prennent en charge que l'ajout de la capacité du noyau `SYS_PTRACE`.

La vidéo suivante montre comment utiliser cette fonctionnalité via le projet Sysdig Falco.

#ContainersFromTheCouch - Résolution des problèmes liés à votre tâche Fargate à l'aide de la fonctionnalité SYS_PTRACE

Le code décrit dans la vidéo précédente se trouve sur GitHub ici.

Utilisez Amazon GuardDuty avec Fargate Runtime Monitoring

Amazon GuardDuty est un service de détection des menaces qui permet de protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, GuardDuty surveille en permanence les différentes sources de journaux et les activités d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

Surveillance du temps d'exécution dans GuardDuty protège les charges de travail exécutées sur Fargate grâce à une surveillance continue AWS journal et activité réseau pour identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un outil léger et entièrement géré GuardDuty agent de sécurité qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l’escalade

des privilèges, l'utilisation d'identifiants exposés ou la communication avec des adresses IP et des domaines malveillants, ainsi que la présence de logiciels malveillants sur vos instances Amazon EC2 et vos charges de travail conteneurisées. Pour plus d'informations, voir GuardDuty Surveillance du temps d'exécution dans le GuardDuty Guide de l'utilisateur.

Considérations relatives à la sécurité Fargate pour Amazon ECS

Chaque tâche dispose d'une capacité d'infrastructure dédiée, car Fargate exécute chaque charge de travail dans un environnement virtuel isolé. Les charges de travail exécutées sur Fargate ne partagent pas les interfaces réseau, le stockage éphémère, le processeur ou la mémoire avec d'autres tâches. Vous pouvez exécuter plusieurs conteneurs au sein d'une même tâche, notamment des conteneurs d'applications et des conteneurs sidecar, ou simplement des sidecars. Un sidecar est un conteneur qui s'exécute parallèlement à un conteneur d'application dans une tâche Amazon ECS. Alors que le conteneur d'applications exécute le code de base de l'application, les processus exécutés dans des sidecars peuvent enrichir l'application. Les sidecars vous aident à séparer les fonctions de l'application dans des conteneurs dédiés, ce qui facilite la mise à jour de certaines parties de votre application.

Les conteneurs qui font partie de la même tâche partagent les ressources pour le type de lancement Fargate, car ces conteneurs s'exécutent toujours sur le même hôte et partagent les ressources de calcul. Ces conteneurs partagent également le stockage éphémère fourni par Fargate. Les conteneurs Linux d'une tâche partagent des espaces de noms réseau, notamment l'adresse IP et les ports réseau. À l'intérieur d'une tâche, les conteneurs appartenant à la tâche peuvent communiquer entre eux via l'hôte local.

L'environnement d'exécution de Fargate vous empêche d'utiliser certaines fonctionnalités du contrôleur prises en charge sur les instances EC2. Tenez compte des éléments suivants lorsque vous concevez des charges de travail qui s'exécutent sur Fargate :

- Aucun conteneur ou accès privilégié : les fonctionnalités telles que les conteneurs ou l'accès privilégié ne sont actuellement pas disponibles sur Fargate. Cela affectera les cas d'utilisation tels que l'exécution de Docker dans Docker.
- Accès limité aux fonctionnalités de Linux : l'environnement dans lequel les conteneurs s'exécutent sur Fargate est verrouillé. Les fonctionnalités Linux supplémentaires, telles que CAP_SYS_ADMIN et CAP_NET_ADMIN, sont limitées afin d'empêcher une escalade des privilèges. Fargate prend en charge l'ajout de la fonctionnalité Linux CAP_SYS_PTRACE aux tâches afin de permettre aux outils d'observabilité et de sécurité déployés dans le cadre de la tâche de surveiller l'application conteneurisée.

- Aucun accès à l'hôte sous-jacent - Ni clients ni AWS les opérateurs peuvent se connecter à un hôte qui exécute les charges de travail des clients. Vous pouvez utiliser ECS Exec pour exécuter des commandes dans ou obtenir un shell vers un conteneur s'exécutant sur Fargate. Vous pouvez utiliser ECS Exec pour collecter des informations de diagnostic pour le débogage. Fargate empêche également les conteneurs d'accéder aux ressources de l'hôte sous-jacent, telles que le système de fichiers, les périphériques, la mise en réseau et l'environnement d'exécution du conteneur.
- Mise en réseau : vous pouvez utiliser des groupes de sécurité et des listes ACL réseau pour contrôler le trafic entrant et sortant. Les tâches Fargate reçoivent une adresse IP depuis le sous-réseau configuré de votre VPC.

Versions de plateforme Fargate pour Amazon ECS

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Il s'agit d'une combinaison de la version Kernel et de la version d'exécution du conteneur. Vous sélectionnez une version de plateforme lorsque vous exécutez une tâche ou lorsque vous créez un service pour gérer un certain nombre de tâches identiques.

De nouvelles révisions des versions de plateforme sont publiées au fil de l'évolution de l'environnement d'exécution, par exemple si des mises à jour, de nouvelles fonctionnalités, des corrections de bugs ou des mises à jour de sécurité sont apportées au noyau ou au système d'exploitation. Une version de plateforme Fargate est mise à jour en effectuant une nouvelle révision de la version de plateforme. Chaque tâche s'exécute sur une révision de version de plateforme au cours de son cycle de vie. Si vous souhaitez utiliser la dernière version de plateforme, vous devez démarrer une nouvelle tâche. Une nouvelle tâche exécutée sur Fargate s'exécute toujours sur la dernière version de plateforme, ce qui garantit que les tâches sont toujours lancées sur une infrastructure sécurisée et corrigée.

Si un problème de sécurité affectant une version existante de la plateforme est détecté, AWS crée une nouvelle révision corrigée de la version de la plateforme et supprime les tâches exécutées sur la révision vulnérable. Dans certains cas, vous pouvez être averti que la résiliation de vos tâches sur Fargate a été planifiée. Pour plus d’informations, consultez Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.

La version de plateforme est spécifiée lors de l’exécution d’une tâche ou du déploiement d’un service.

Tenez compte des éléments suivants lorsque vous spécifiez une version de plateforme :

- Vous pouvez spécifier un numéro de version spécifique, par exemple `1.4.0` ou `LATEST`.

  La DERNIÈRE version de plateforme Linux est `1.4.0`.

  La DERNIÈRE version de plateforme Windows est `1.0.0`.

- Si vous souhaitez mettre à jour la version de plateforme d'un service, créez un déploiement. Par exemple, supposons que vous ayez un service qui exécute des tâches sur la version de plateforme Linux `1.3.0`. Pour modifier le service afin d'exécuter des tâches sur la version de plateforme Linux `1.4.0`, mettez à jour votre service et spécifiez une nouvelle version de plateforme. Vos tâches sont redéployées avec la dernière version de plateforme et la dernière révision de la version de plateforme. Pour de plus amples informations sur les déploiements, veuillez consulter Services Amazon ECS.
- Si votre service est mis à l'échelle sans que la version de plateforme soit mise à jour, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel de ce service. Par exemple, supposons que vous ayez un service qui exécute des tâches sur la version de plateforme Linux `1.3.0`. Si vous augmentez le nombre souhaité de services, le planificateur de services lance les nouvelles tâches en utilisant la dernière version de plateforme, révision de la version de plateforme `1.3.0`.
- Les nouvelles tâches s'exécutent toujours sur la dernière révision d'une version de plateforme. Cela garantit que les tâches sont toujours exécutées sur une infrastructure sécurisée et mise à jour.
- Les numéros de version de plateforme pour les conteneurs Linux et les conteneurs Windows sur Fargate sont indépendants. Par exemple, le comportement, les fonctionnalités et les logiciels utilisés dans la version de plateforme `1.0.0` pour les conteneurs Windows sur Fargate ne sont pas comparables à ceux de la version de plateforme `1.0.0` pour les conteneurs Linux sur Fargate.
- Ce qui suit s'applique aux versions de plateforme Windows Fargate.

  Les images de conteneur Microsoft Windows Server doivent être créées à partir d'une version spécifique de Windows Server. Vous devez sélectionner la même version de Windows Server dans la `platformFamily` lorsque vous exécutez une tâche ou créez un service correspondant à l'image du conteneur Windows Server. En outre, vous pouvez fournir une correspondance `operatingSystemFamily` dans la définition de tâche afin d'éviter que les tâches ne soient exécutées sur la mauvaise version de Windows. Pour plus d'informations, veuillez consulter Correspondance de la version d'hôte de conteneur avec les versions des images de conteneur sur le site Web de Microsoft Learn.

Migration vers la version de plateforme Linux 1.4.0 sur Amazon ECS

Tenez compte des éléments suivants lors de la migration de vos tâches Amazon ECS sur Fargate à partir de la version `1.0.0`, `1.1.0`, `1.2.0` ou `1.3.0` de la plateforme vers la version `1.4.0`. Il est recommandé de vérifier que votre tâche fonctionne correctement sur la version de plateforme `1.4.0` avant de procéder à la migration des tâches.

- Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version de plateforme 1.4.0, toutes les tâches Amazon ECS sur Fargate reçoivent une interface réseau Elastic (appelée ENI de tâche) unique et tout le trafic réseau transite par cette ENI au sein de votre VPC. Le trafic est visible dans les journaux de flux de votre VPC. Pour plus d'informations, consultez Options de mise en réseau des tâches Amazon ECS pour Fargate.
- Si vous utilisez des points de terminaison de VPC d'interface, tenez compte des éléments suivants.
  - Pour les images de conteneur hébergées avec Amazon ECR, vous avez besoin des points de terminaison suivants. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
    - com.amazonaws.***region***.ecr.dkr Point de terminaison Amazon ECR VPC
    - com.amazonaws.***region***.ecr.api Point de terminaison Amazon ECR VPC
    - Point de terminaison de la passerelle Amazon S3
  - Lorsque votre définition de tâche fait référence aux secrets Secrets Manager pour récupérer des données sensibles pour vos conteneurs, vous devez créer les points de terminaison de VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
  - Lorsque votre définition de tâche fait référence aux paramètres du magasin de paramètres Systems Manager pour récupérer des données sensibles pour vos conteneurs, vous devez créer les points de terminaison de VPC d'interface pour Systems Manager. Pour plus d'informations, consultez la section Amélioration de la sécurité des instances EC2 à l'aide des points de terminaison de VPC pour Systems Manager dans le Guide de l'utilisateur AWS Systems Manager .
  - Le groupe de sécurité pour l'interface réseau Elastic (ENI) associé à votre tâche a besoin des règles du groupe de sécurité pour autoriser le trafic entre la tâche et les points de terminaison de VPC.

Journal des modifications de la version de plateforme Linux Fargate

Les versions de plateforme Linux suivantes sont proposées. Pour obtenir des informations sur l'obsolescence des versions de plateforme, consultez AWS Dépréciation de la version de la plateforme Fargate Linux.

1.4.0

Voici le journal des modifications de la version `1.4.0` de la plateforme.

- Depuis le 5 novembre 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate en utilisant la version `1.4.0` de la plateforme peut utiliser les fonctions suivantes :
  - Lorsque vous utilisez Secrets Manager pour stocker des données sensibles, vous pouvez injecter une clé JSON spécifique ou une version spécifique d'un secret en tant que variable d'environnement ou dans une configuration de journal. Pour plus d’informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.
  - Spécifiez les variables d'environnement en bloc à l'aide du paramètre de définition de conteneur `environmentFiles`. Pour plus d’informations, consultez Transmission d’une variable d’environnement individuelle à un conteneur Amazon ECS.
  - Les tâches exécutées dans un VPC et un sous-réseau activés pour IPv6 se verront attribuer à la fois une adresse IPv4 et une adresse IPv6 privées. Pour plus d’informations, consultez la section Options de mise en réseau des tâches Amazon ECS pour Fargate.
  - Le point de terminaison de métadonnées de tâche version 4 fournit des métadonnées supplémentaires sur votre tâche et conteneur, y compris le type de lancement de tâche, l'Amazon Resource Name (ARN) du conteneur, ainsi que le pilote de journal et les options de pilote de journal utilisés. Lors de l'interrogation du point de terminaison `/stats`, vous recevez également des statistiques de débit réseau pour vos conteneurs. Pour plus d’informations, consultez la section Point de terminaison des métadonnées de tâche, version 4.
- Depuis le 30 juillet 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate en utilisant la version `1.4.0` de la plateforme peut acheminer le trafic UDP à l'aide d'un Network Load Balancer vers Amazon ECS sur les tâches Fargate. Pour plus d’informations, consultez Utilisation de l’équilibrage de charge pour répartir le trafic des services Amazon ECS.
- À compter du 28 mai 2020, toute nouvelle tâche Amazon ECS lancée sur Fargate à l'aide de la version de la plateforme `1.4.0` verra son stockage éphémère chiffré avec un AES-256 algorithme de chiffrement à l'aide d'un AWS clé de chiffrement détenue. Pour plus d’informations, consultez

Stockage éphémère des tâches Fargate pour Amazon ECS et Options de stockage pour les tâches Amazon ECS.

- Ajout de la prise en charge de l'utilisation des volumes de système de fichiers Amazon EFS pour le stockage des tâches permanentes. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.
- Le stockage des tâches éphémères est désormais d'au moins 20 Go pour chaque tâche. Pour plus d'informations, consultez Stockage éphémère des tâches Fargate pour Amazon ECS.
- Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version 1.4.0 de la plateforme, toutes les tâches Fargate reçoivent une interface réseau Elastic unique (appelée ENI de tâche), et tout le trafic réseau passe par cette ENI au sein du VPC et vous est accessible via les journaux de vos flux VPC. Pour plus d'informations sur la mise en réseau pour le type de lancement Amazon EC2, consultez la section Options de mise en réseau des tâches Amazon ECS pour EC2. Pour plus d'informations sur la mise en réseau pour Fargate, consultez la section Options de mise en réseau des tâches Amazon ECS pour Fargate.
- Les ENI des tâches incluent désormais la prise en charge des trames jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficacité. La prise en charge des trames jumbo limite la surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo, telles que tout le trafic qui reste dans votre VPC.
- CloudWatch Container Insights inclura des mesures de performance du réseau pour les tâches Fargate. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.
- Ajout de la prise en charge du point de terminaison de métadonnées de tâche version 4, qui fournit des informations supplémentaires pour vos tâches Fargate, y compris les statistiques réseau pour la tâche et la zone de disponibilité dans laquelle la tâche s'exécute. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4 et Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate.
- Ajout de la prise en charge du paramètre `SYS_PTRACE` Linux dans les définitions de conteneur. Pour plus d'informations, consultez Paramètres Linux.
- L'agent de conteneur Fargate remplace l'agent de conteneur Amazon ECS pour toutes les tâches Fargate. Généralement, cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent.

- L'environnement d'exécution du conteneur utilise maintenant Containerd au lieu de Docker. Dans la plupart des cas, cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. Vous remarquerez que certains messages d'erreur provenant de l'environnement d'exécution du conteneur mentionnent de moins en moins Docker et de en plus des erreurs générales. Pour plus d'informations, consultez la section Messages d'erreur liés aux tâches Amazon ECS arrêtées.
- Basée sur Amazon Linux 2.

AWS Dépréciation de la version de la plateforme Fargate Linux

Le tableau suivant répertorie les versions de la plate-forme Linux qui AWS Fargate est obsolète ou sa mise en dépréciation est prévue. Ces versions de plateforme restent disponibles jusqu'à la date d'obsolescence publiée.

Une date de mise à jour forcée est fournie pour chaque version de plateforme dont l'obsolescence est programmée. À la date de mise à jour forcée, tout service utilisant la version de plateforme `LATEST` qui est indiquée comme une version dont l'obsolescence est programmée sera mise à jour à l'aide de l'option Forcer le nouveau déploiement. Lorsque le service est mis à jour à l'aide de l'option Forcer le nouveau déploiement, toutes les tâches exécutées sur une version de plateforme dont l'obsolescence est programmée sont arrêtées et de nouvelles tâches sont lancées à l'aide de la version de plateforme que la balise `LATEST` indique à ce moment. Les tâches ou services autonomes avec un jeu de version de plateforme explicite ne sont pas affectés par la date de mise à jour forcée.

Pour plus d'informations sur la migration vers la dernière version de plateforme, consultez la section Migration vers la version de plateforme Linux 1.4.0 sur Amazon ECS.

Une fois qu'une version de plateforme atteint sa date d'obsolescence, la version de plateforme ne sera plus disponible pour de nouvelles tâches ou services. Toutes les tâches ou services autonomes qui utilisent explicitement une version de plateforme obsolète continueront d'utiliser cette version de plateforme jusqu'à ce que les tâches soient arrêtées. Après la date d'obsolescence, une version de plateforme obsolète ne recevra plus de mises à jour de sécurité ou de corrections de bugs.

Version de plateforme	Date de mise à jour forcée	Date d'obsolescence
1.0.0	26 octobre 2020	14 décembre 2020
1.1.0	26 octobre 2020	14 décembre 2020

Version de plateforme	Date de mise à jour forcée	Date d'obsolescence
1.2.0	26 octobre 2020	14 décembre 2020
1.3.0		15 juin 2026

Pour obtenir des informations sur les versions de plateforme actuelles, consultez Versions de plateforme Fargate pour Amazon ECS.

Journal des modifications des versions obsolètes de Fargate Linux

1.3.0

Voici le journal des modifications de la version `1.3.0` de la plateforme.

- Depuis le 30 septembre 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `awsfirelens`. Configurez le FireLens pour qu'Amazon ECS utilise les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou AWS Destination APN (Partner Network) pour le stockage et l'analyse des journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.
- Ajout du recyclage de tâche pour les tâches Fargate, qui est le processus d'actualisation des tâches qui font partie d'un service Amazon ECS service. Pour plus d'informations, Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.
- Depuis le 27 mars 2019, toute nouvelle tâche Fargate lancée peut utiliser des paramètres de définition de tâche supplémentaires qui vous permettent de définir une configuration proxy, des dépendances de démarrage et d'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de démarrage par conteneur. Pour plus d'informations, consultez Configuration du proxy, Dépendances du conteneur et Temporisations de conteneurs.
- À compter du 2 avril 2019, toute nouvelle tâche Fargate lancée permet d'injecter des données sensibles dans vos conteneurs en stockant vos données sensibles dans AWS Secrets Manager secrets ou AWS Systems Manager Paramètre Stockez les paramètres, puis référencez-les dans la définition de votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.
- Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le référencement des données sensibles dans la configuration de journal d'un conteneur à l'aide du paramètre de définition de conteneur `secretOptions`. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

- Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `splunk` en plus du pilote de journal `awslogs`. Pour plus d'informations, consultez Stockage et journalisation.
- À compter du 9 juillet 2019, toute nouvelle tâche Fargate lancée prend en charge CloudWatch Informations sur les conteneurs. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.
- Depuis le 3 décembre 2019, le fournisseur de capacité Fargate Spot est pris en charge. Pour plus d'informations, consultez Clusters Amazon ECS pour Fargate.
- Basée sur Amazon Linux 2.

1.2.0

Voici le journal des modifications de la version `1.2.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.2.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, consultez AWS Dépréciation de la version de la plateforme Fargate Linux.

- Ajout du support pour l'authentification dans le registre privé à l'aide de AWS Secrets Manager. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

1.1.0

Voici le journal des modifications de la version `1.1.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.1.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, veuillez consulter AWS Dépréciation de la version de la plateforme Fargate Linux.

- Ajout de la prise en charge du point de terminaison des métadonnées de tâches Amazon ECS. Pour plus d'informations, consultez Métadonnées de tâches Amazon ECS disponibles pour les tâches sur Fargate.
- Ajout de la prise en charge des surveillances de l'état Docker dans les définitions de conteneur. Pour plus d'informations, consultez Surveillance de l'état.
- Ajout de la prise en charge de la découverte de service Amazon ECS service. Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.

1.0.0

Voici le journal des modifications de la version `1.0.0` de la plateforme.

> **Note**
>
> La version de plateforme `1.0.0` n'est plus disponible. Pour obtenir des informations sur l'obsolescence des versions de plateforme, veuillez consulter AWS Dépréciation de la version de la plateforme Fargate Linux.

- Basée sur Amazon Linux 2017.09.
- Première version.

Journal des modifications de la version de plateforme Windows Fargate

Les versions de plateforme pour les conteneurs Windows suivantes sont proposées.

1.0.0

Voici le journal des modifications de la version `1.0.0` de la plateforme.

- Version initiale pour la prise en charge des systèmes d'exploitation Microsoft Windows Server suivants :
  - Windows Server 2019 Full
  - Windows Server 2019 Core
  - Windows Server 2022 Full
  - Windows Server 2022 Core

Considérations sur les conteneurs Windows sur Fargate pour Amazon ECS

Voici les différences et les considérations à prendre en compte lorsque vous exécutez des conteneurs Windows sur AWS Fargate.

Si vous devez exécuter des tâches sur des conteneurs Linux et Windows, vous devez créer des définitions de tâches distinctes pour chaque système d'exploitation.

AWS gère la gestion des licences du système d'exploitation, vous n'avez donc pas besoin de licences Microsoft Windows Server supplémentaires.

Conteneurs Windows allumés AWS Fargate est compatible avec les systèmes d'exploitation suivants :

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Conteneurs Windows allumés AWS Fargate prend en charge le pilote awslogs. Pour plus d'informations, consultez the section called “Envoyer les journaux à CloudWatch ”.

Les fonctionnalités suivantes ne sont pas prises en charge sur les conteneurs Windows sur Fargate :

- Amazon FSx
- Jonction ENI
- gMSA pour les conteneurs Windows
- Les intégrations du service App Mesh et du proxy pour les tâches
- L'intégration du routeur journal Firelens pour les tâches
- Les volumes EFS
- Volumes EBS
- Les paramètres de définition de tâche suivants :
  - `maxSwap`
  - `swappiness`
  - `environmentFiles`

  - `ephemeralStorage`
- Le fournisseur de capacités Fargate Spot
- Les volumes d'images

  L'option Dockerfile `volume` est ignorée. Au lieu de cela, utilisez des montages liés dans votre définition de tâche. Pour plus d'informations, consultez Utilisation de montages liés avec Amazon ECS.
- Les paramètres d'UC et de mémoire au niveau des tâches sont ignorés pour les conteneurs Windows. Nous vous recommandons de spécifier des ressources de niveau conteneur pour les conteneurs Windows.
- Mémoire pour la tâche
- Réservation de mémoire sur les conteneurs
- Politiques de redémarrage relatives aux conteneurs
- Vous ne pouvez pas mettre à jour la famille de plateformes d'un service.

Comportement d'extraction d'image de conteneur Linux sur Fargate pour Amazon ECS

Chaque tâche Fargate s'exécute sur sa propre instance à usage unique et à locataire unique. Lorsque vous exécutez des conteneurs Linux sur Fargate, les images de conteneurs ou les couches d'images de conteneurs ne sont pas mises en cache sur l'instance. Par conséquent, pour chaque image de conteneur définie dans la tâche, l'image de conteneur complète doit être extraite du registre d'images de conteneur pour chaque tâche Fargate. Le temps nécessaire pour extraire les images est directement corrélé au temps nécessaire pour démarrer une tâche Fargate.

Tenez compte des points suivants pour optimiser le temps d'extraction de l'image.

Proximité des images du conteneur

Pour réduire le temps nécessaire au téléchargement des images du conteneur, localisez les données le plus près possible du calcul. Extraire une image de conteneur sur Internet ou sur Régions AWS peut avoir un impact sur le temps de téléchargement. Nous vous recommandons de stocker l'image du conteneur dans la même région que celle où la tâche sera exécutée. Si vous stockez l'image du conteneur dans Amazon ECR, utilisez un point de terminaison de VPC d'interface pour réduire davantage le temps d'extraction de l'image. Pour de plus amples

informations, consultez la section Points de terminaison de VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon VPC.

Réduction de la taille de l'image du conteneur

La taille d'une image de conteneur a un impact direct sur le temps de téléchargement. La réduction de la taille de l'image du conteneur ou du nombre de couches de l'image du conteneur peut réduire le temps nécessaire au téléchargement d'une image. Les images de base légères (telles que l'image de conteneur minimale Amazon Linux 2023) peuvent être nettement plus petites que celles basées sur des images de base de système d'exploitation traditionnelles. Pour plus d'informations sur l'image minimale, consultez la section Image de conteneur minimale AL2023 dans le Guide de l'utilisateur Amazon Linux 2023.

Algorithmes de compression alternatifs

Les couches d'images de conteneurs sont souvent compressées lorsqu'elles sont transférées vers un registre d'images de conteneurs. La compression de la couche d'image du conteneur réduit la quantité de données qui doivent être transférées sur le réseau et stockées dans le registre des images du conteneur. Une fois qu'une couche d'image de conteneur a été téléchargée sur une instance par le moteur d'exécution du conteneur, cette couche est décompressée. L'algorithme de compression utilisé et le nombre de vCPU disponibles au moment de l'exécution ont un impact sur le temps nécessaire pour décompresser l'image du conteneur. Sur Fargate, vous pouvez augmenter la taille de la tâche ou tirer parti de l'algorithme de compression zstd, plus performant, pour réduire le temps de décompression. Pour plus d'informations, voir std sur GitHub. Pour plus d'informations sur la façon d'implémenter les images pour Fargate, voir Réduire AWS Fargate Temps de démarrage avec les images de conteneurs compressées zstd.

Chargement différé d'images de conteneurs

Pour les images de conteneur de grande taille (> 250 Mo), il peut être préférable de procéder à un chargement différé plutôt que de télécharger l'intégralité de l'image. Sur Fargate, vous pouvez utiliser Seekable OCI (SOCI) pour procéder au chargement différé d'une image de conteneur à partir d'un registre d'images de conteneur. Pour plus d'informations, voir soci-snapshotter sur GitHub et Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI).

Comportement d’extraction d’image de conteneur relatif aux conteneurs Windows sur Fargate pour Amazon ECS

Fargate Windows met en cache l’image de base Server Core fournie par Microsoft pour le mois en cours et le mois précédent. Ces images correspondent à KB/Build nombre de patchs mis à jour chaque Patch Tuesday. Par exemple, le 9 avril 2024, Microsoft a publié KB5036896 (17763.5696) pour Windows Server 2019. Le KB précédent, daté du 12 mars 2024, était le KB5035849 (17763.5576). Ainsi, pour les plateformes `WINDOWS_SERVER_2019_CORE` et `WINDOWS_SERVER_2019_FULL`, les images de conteneurs suivantes ont été mises en cache :

- `mcr.microsoft.com/windows/servercore:ltsc2019`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5696`
- `mcr.microsoft.com/windows/servercore:10.0.17763.5576`

De plus, le 9 avril 2024, Microsoft a publié le KB5036909 (20348.2402) pour Windows Server 2022. Le KB précédent, daté du 12 mars 2024, était le KB5035857 (20348.2340). Ainsi, pour les plateformes `WINDOWS_SERVER_2022_CORE` et `WINDOWS_SERVER_2022_FULL`, les images de conteneurs suivantes ont été mises en cache :

- `mcr.microsoft.com/windows/servercore:ltsc2022`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2402`
- `mcr.microsoft.com/windows/servercore:10.0.20348.2340`

Stockage éphémère des tâches Fargate pour Amazon ECS

Une fois provisionnée, chaque tâche Amazon ECS hébergée sur des conteneurs Linux sur AWS Fargate reçoit le stockage éphémère suivant pour les montages de reliure. Ils peuvent être montés et partagés entre les conteneurs en utilisant les paramètres `volumes`, `mountPoints` et `volumesFrom` dans la définition de tâche. Ceci n'est pas pris en charge pour les conteneurs Windows sur AWS Fargate.

Versions de la plateforme de conteneurs Linux Fargate

Version 1.4.0 ou ultérieure

Par défaut, les tâches Amazon ECS sur Fargate utilisant la version `1.4.0` ou ultérieure de la plateforme reçoivent un minimum de 20 Gio de stockage éphémère. La quantité totale de stockage éphémère peut être augmentée, jusqu'à un maximum de 200 Gio. Vous pouvez effectuer cette opération en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche.

Les images de conteneur extraites, compressées et non compressées de la tâche sont stockées sur le stockage éphémère. Pour déterminer la quantité totale de stockage éphémère que votre tâche doit utiliser, vous devez soustraire la quantité de stockage utilisée par votre image de conteneur de la quantité totale de stockage éphémère alloué à votre tâche.

Pour les tâches qui utilisent la version de la plateforme `1.4.0` ou versions ultérieures qui sont lancées le 28 mai 2020 ou plus tard, le stockage éphémère est crypté avec un AES-256 algorithme de chiffrement. Cet algorithme utilise AWS clé de chiffrement que vous possédez, ou vous pouvez créer votre propre clé gérée par le client. Pour plus d'informations, voir Clés gérées par le client pour AWS Fargate stockage éphémère.

Pour les tâches utilisant la version `1.4.0` ou ultérieure de la plateforme qui sont lancées le 18 novembre 2022 ou plus tard, l'utilisation du stockage éphémère est signalée sur le point de terminaison des métadonnées de la tâche. Dans le cadre de vos tâches, vos applications peuvent interroger le point de terminaison des métadonnées des tâches, version 4, pour connaître la taille réservée au stockage éphémère et la quantité utilisée.

De plus, la taille réservée au stockage éphémère et la quantité utilisée sont envoyées à Amazon CloudWatch Container Insights si vous activez Container Insights.

> **Note**
>
> Fargate réserve de l'espace sur le disque. Il n'est utilisé que par Fargate. Vous n'êtes pas facturé pour cela. Cela n'apparaît pas dans ces statistiques. Toutefois, vous pouvez voir ce stockage supplémentaire dans d'autres outils tels que `df`.

Version 1.3.0 ou antérieure

Pour les tâches Amazon ECS sur Fargate utilisant la version `1.3.0` ou antérieure de la plateforme, chaque tâche reçoit le magasin éphémère suivant.

- 10 Go de stockage de couche Docker

> **Note**
>
> Ce volume inclut les artefacts d'image de conteneur compressée et non compressée.

- 4 Go supplémentaires pour les montages de volume. Ils peuvent être montés et partagés entre les conteneurs en utilisant les paramètres `volumes`, `mountPoints` et `volumesFrom` dans la définition de tâche.

Clés gérées par le client pour AWS Fargate stockage éphémère pour Amazon ECS

AWS Fargate prend en charge les clés gérées par les clients pour chiffrer les données relatives aux tâches Amazon ECS stockées dans un stockage éphémère afin d'aider les clients sensibles à la réglementation à respecter leurs politiques de sécurité internes. Les clients continuent de bénéficier des avantages sans serveur de Fargate, tout en offrant une meilleure visibilité sur le chiffrement du stockage autogéré aux auditeurs de conformité. Alors que Fargate a Fargate-managed chiffrement de stockage éphémère par défaut, les clients peuvent également utiliser leurs propres clés autogérées pour chiffrer des données sensibles telles que des informations financières ou liées à la santé.

Vous pouvez importer vos propres clés dans AWS KMS ou créez les clés dans AWS KMS. Ces clés autogérées sont stockées dans AWS KMS et exécuter de manière standard AWS KMS les actions du cycle de vie telles que la rotation, la désactivation et la suppression. Vous pouvez auditer l'accès et l'utilisation des clés dans CloudTrail journaux.

Par défaut, la clé KMS prend en charge 50 000 autorisations par clé. Fargate utilise un seul AWS KMS attribution par tâche clé gérée par client, de sorte qu'il prend en charge jusqu'à 50 000 tâches simultanées pour une clé. Si vous souhaitez augmenter ce nombre, vous pouvez demander une augmentation de limite, qui est approuvée au cas par cas.

Fargate ne facture aucun supplément pour l’utilisation des clés gérées par le client. Seul le prix standard vous est facturé pour l'utilisation AWS KMS clés pour le stockage et les demandes d'API.

Rubriques

- Création d’une clé de chiffrement pour le stockage éphémère Fargate pour Amazon ECS
- Gestion AWS KMS clés pour le stockage éphémère Fargate pour Amazon ECS

Création d'une clé de chiffrement pour le stockage éphémère Fargate pour Amazon ECS

Créer une clé gérée par le client pour le chiffrement des données stockées dans le stockage éphémère Fargate.

> **Note**
>
> Le chiffrement du stockage éphémère Fargate avec des clés gérées par le client n'est pas disponible pour les clusters de tâches Windows.
> Le chiffrement du stockage éphémère Fargate avec des clés gérées par le client n'est pas disponible sur les `platformVersions` antérieures à la version `1.4.0`.
> Fargate réserve de l'espace sur un espace de stockage éphémère qui n'est utilisé que par Fargate, et cet espace ne vous est pas facturé. L'allocation peut différer des tâches de clés non gérées par le client, mais l'espace total reste le même. Vous pouvez visualiser cette modification dans des outils tels que `df`.
> Multi-Region les clés ne sont pas prises en charge pour le stockage éphémère Fargate.
> Les alias de clé KMS ne sont pas pris en charge pour le stockage éphémère Fargate.

Pour créer une clé gérée par le client (CMK) afin de chiffrer le stockage éphémère de Fargate dans AWS KMS, procédez comme suit.

1. Accédez à https://console.aws.amazon.com/kms.
2. Suivez les instructions de la section Création de clés dans le Guide du développeur AWS Key Management Service.
3. Lors de la création de votre AWS KMS clé, assurez-vous de fournir un service Fargate pertinent AWS KMS autorisations d'exploitation dans les politiques clés. Les opérations d'API suivantes doivent être autorisées dans la politique pour pouvoir utiliser votre clé gérée par le client avec vos ressources de cluster Amazon ECS.

   - `kms:GenerateDataKeyWithoutPlainText` - Appelez `GenerateDataKeyWithoutPlainText` pour générer une clé de données cryptée à partir du AWS KMS clé.
   - `kms:CreateGrant` : ajoute un octroi à une clé gérée par le client. Permet de contrôler l'accès à une donnée AWS KMS clé, qui permet d'accéder aux opérations d'octroi requises par Amazon ECS Fargate. Pour plus d'informations à propos de l'Utilisation des octrois, consultez

le Guide du développeur AWS Key Management Service. Cela permet à Amazon ECS Fargate d'effectuer les opérations suivantes :

- Appel `Decrypt` pour AWS KMS pour obtenir la clé de chiffrement permettant de déchiffrer les données de stockage éphémères.
- Configurer un principal sortant pour permettre au service de `RetireGrant`.

- `kms:DescribeKey` : fournit les détails de la clé gérée par le client pour permettre à Amazon ECS de valider la clé si elle est symétrique et activée.

L'exemple suivant montre un AWS KMS politique de clé que vous appliqueriez à la clé cible pour le chiffrement. Pour utiliser les exemples d'instructions de politique, remplacez les *`user input placeholders`* informations par les vôtres. Comme toujours, configurez uniquement les autorisations dont vous avez besoin, mais vous devrez fournir AWS KMS avec des autorisations pour au moins un utilisateur afin d'éviter les erreurs.

```
{
      "Sid": "Allow generate data key access for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:GenerateDataKeyWithoutPlaintext"
      ],
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow grant creation permission for Fargate tasks.",
      "Effect": "Allow",
      "Principal": { "Service":"fargate.amazonaws.com" },
      "Action": [
        "kms:CreateGrant"
      ],
```

```
      "Condition": {
        "StringEquals": {
          "kms:EncryptionContext:aws:ecs:clusterAccount": [
            "customerAccountId"
          ],
          "kms:EncryptionContext:aws:ecs:clusterName": [
             "clusterName"
          ]
        },
       "ForAllValues:StringEquals": {
          "kms:GrantOperations": [
             "Decrypt"
          ]
       }
      },
      "Resource": "*"
    },
    {
      "Sid": "Allow describe key permission for cluster operator - CreateCluster
 and UpdateCluster.",
      "Effect": "Allow",
      "Principal": { "AWS":"arn:aws:iam::customerAccountId:role/customer-chosen-
role" },
      "Action": [
        "kms:DescribeKey"
      ],
      "Resource": "*"
    }
```

Les tâches Fargate utilisent les clés de contexte de chiffrement `aws:ecs:clusterAccount` et `aws:ecs:clusterName` pour les opérations cryptographiques effectuées avec la clé. Les clients doivent ajouter ces autorisations pour restreindre l'accès à un compte spécifique and/or grappe. Utilisez le nom du cluster et non l'ARN lorsque vous spécifiez le cluster.

Consultez Contexte de chiffrement dans le AWS KMS guide du développeur pour en savoir plus.

Lors de la création ou de la mise à jour d'un cluster, vous avez la possibilité d'utiliser la clé de condition `fargateEphemeralStorageKmsKeyId`. Cette clé de condition permet aux clients d'exercer un contrôle plus précis sur les politiques IAM. Les mises à jour de configuration `fargateEphemeralStorageKmsKeyId` ne prennent effet que sur les nouveaux déploiements de service.

Voici un exemple de la manière dont les clients peuvent accorder des autorisations uniquement à un ensemble spécifique de personnes approuvées AWS KMS clés.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
      ],
      "Resource": "*",
      "Condition": {
        "StringEquals": {
          "ecs:fargate-ephemeral-storage-kms-key": "a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
      }
    }
  ]
}
```

Voici un exemple pour refuser les tentatives de suppression AWS KMS clés déjà associées à un cluster.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": {
    "Effect": "Deny",
    "Action": [
        "ecs:CreateCluster",
        "ecs:UpdateCluster"
    ],
    "Resource": "*",
```

```
    "Condition": {
      "Null": {
        "ecs:fargate-ephemeral-storage-kms-key": "true"
      }
    }
  }
}
```

Les clients peuvent voir si leurs tâches non gérées ou leurs tâches de service sont chiffrées à l'aide de la clé en utilisant AWS CLI `describe-tasks`, `describe-cluster`, ou `describe-services` commandes.

Pour plus d'informations, consultez la section Clés de condition pour AWS KMS dans le Guide du développeur AWS KMS.

Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Choisissez Clusters dans le volet de navigation et soit Créer un cluster, ou choisissez un cluster existant. Pour un cluster existant, choisissez Mettre à jour le cluster.
3. Dans le cadre de Chiffrement section du flux de travail, vous aurez la possibilité de sélectionner votre AWS KMS touche en dessous Stockage géré et Stockage éphémère Fargate. Vous pouvez également choisir Créez un AWS KMS clé d'ici.
4. Choisissez Créer une fois que vous avez fini de créer votre cluster ou Mettre à jour, si vous mettiez à jour un cluster existant.

AWS CLI

Voici un exemple de création d'un cluster et de configuration de votre stockage éphémère Fargate à l'aide du AWS CLI (remplacez le *red* valeurs (avec les vôtres) :

```
aws ecs create-cluster --cluster clusterName \
--configuration '{"managedStorageConfiguration":
{"fargateEphemeralStorageKmsKeyId":"arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"}}'
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-west-2:012345678901:cluster/clusterName",
```

```
        "clusterName": "clusterName",
        "configuration": {
            "managedStorageConfiguration": {
                "fargateEphemeralStorageKmsKeyId": "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
            }
        },
        "status": "ACTIVE",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": []
    },
    "clusterCount": 5
}
```

CloudFormation

Voici un exemple de modèle de création d'un cluster et de configuration de votre stockage éphémère Fargate à l'aide du CloudFormation (remplacez le *red* valeurs (avec les vôtres) :

```
AWSTemplateFormatVersion: 2010-09-09
Resources:
  MyCluster:
    Type: AWS::ECS::Cluster
    Properties:
      ClusterName: "clusterName"
      Configuration:
        ManagedStorageConfiguration:
          FargateEphemeralStorageKmsKeyId: "arn:aws:kms:us-
west-2:012345678901:key/a1b2c3d4-5678-90ab-cdef-EXAMPLE11111"
```

Gestion AWS KMS clés pour le stockage éphémère Fargate pour Amazon ECS

Après avoir créé ou importé votre AWS KMS clé pour crypter votre stockage éphémère Fargate, vous le gérez comme vous le feriez pour n'importe quel autre AWS KMS clé.

Rotation automatique de AWS KMS clés

Vous pouvez activer la rotation automatique des clés ou les faire pivoter manuellement. La rotation automatique des clés fait tourner la clé pour vous chaque année en générant de nouveaux éléments cryptographiques pour la clé. AWS KMS enregistre également toutes les versions précédentes du matériel cryptographique, de sorte que vous serez en mesure de déchiffrer toutes les données utilisant les versions de clé précédentes. Aucun matériau pivoté ne sera supprimé par AWS KMS jusqu'à ce que vous supprimiez la clé.

La rotation automatique des clés est facultative et peut être activée ou désactivée à tout moment.

Désactivation ou révocation AWS KMS clés

Si vous désactivez une clé gérée par le client dans AWS KMS, cela n'a aucun impact sur l'exécution des tâches, qui continuent de fonctionner tout au long de leur cycle de vie. Si une nouvelle tâche utilise la clé désactivée ou révoquée, elle échoue, car elle ne peut pas accéder à la clé. Vous devez définir un CloudWatch alarme ou similaire pour vous assurer qu'une clé désactivée n'est jamais nécessaire pour déchiffrer des données déjà cryptées.

Suppression AWS KMS clés

La suppression de clés doit toujours être envisagée en dernier recours et ne doit être effectuée que si vous êtes certain que la clé supprimée ne sera plus jamais nécessaire. Les nouvelles tâches qui tentent d'utiliser la clé supprimée échoueront car elles ne pourront pas y accéder. AWS KMS conseille de désactiver une clé au lieu de la supprimer. Si vous pensez qu'il est nécessaire de supprimer une clé, nous vous suggérons de la désactiver d'abord et de définir une CloudWatch alarme pour s'assurer qu'elle n'est pas nécessaire. Si vous supprimez une clé, AWS KMS fournit au moins sept jours pour vous faire changer d'avis.

Audit AWS KMS accès par clé

Vous pouvez utiliser CloudTrail journaux pour auditer l'accès à votre AWS KMS clé. Vous pouvez vérifier AWS KMS opérations `CreateGrant`, `GenerateDataKeyWithoutPlaintext`, et `Decrypt`. Ces opérations montrent également `aws:ecs:clusterAccount` et `aws:ecs:clusterName` dans le cadre du `EncryptionContext` connecté CloudTrail.

Voici des exemples CloudTrail événements pour `GenerateDataKeyWithoutPlaintext`, `GenerateDataKeyWithoutPlaintext (DryRun)`, `CreateGrant`, `CreateGrant (DryRun)`, et `RetireGrant` (remplacez le *red* valeurs avec les vôtres).

GenerateDataKeyWithoutPlaintext


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "numberOfBytes": 64,
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ebs:id": "vol-xxxxxxx",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

GenerateDataKeyWithoutPlaintext (DryRun)


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "GenerateDataKeyWithoutPlaintext",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "dryRun": true,
        "numberOfBytes": 64,
        "encryptionContext": {
            "aws:ecs:clusterAccount": "account-id",
            "aws:ecs:clusterName": "cluster-name"
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": true,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
```

```
}
```

CreateGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "ec2-frontend-api.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:13Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "ec2-frontend-api.amazonaws.com",
    "userAgent": "ec2-frontend-api.amazonaws.com",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ebs:id": "vol-xxxx",
                "aws:ecs:clusterName": "cluster-name"
            }
        },
        "retiringPrincipal": "ec2.us-west-2.amazonaws.com"
    },
    "responseElements": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
```

```
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

CreateGrant (DryRun)

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "fargate.amazonaws.com"
    },
    "eventTime": "2024-04-23T18:08:11Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "CreateGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "fargate.amazonaws.com",
    "userAgent": "fargate.amazonaws.com",
    "errorCode": "DryRunOperationException",
    "errorMessage": "The request would have succeeded, but the DryRun option is
 set.",
    "requestParameters": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111",
        "granteePrincipal": "fargate.us-west-2.amazonaws.com",
        "dryRun": true,
        "operations": [
            "Decrypt"
        ],
        "constraints": {
            "encryptionContextSubset": {
                "aws:ecs:clusterAccount": "account-id",
                "aws:ecs:clusterName": "cluster-name"
            }
```

```
        }
    },
    "responseElements": null,
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

RetireGrant

```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "AWSService",
        "invokedBy": "AWS Internal"
    },
    "eventTime": "2024-04-20T18:37:38Z",
    "eventSource": "kms.amazonaws.com",
    "eventName": "RetireGrant",
    "awsRegion": "us-west-2",
    "sourceIPAddress": "AWS Internal",
    "userAgent": "AWS Internal",
    "requestParameters": null,
    "responseElements": {
        "keyId": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
    },
    "additionalEventData": {
        "grantId":
 "e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855"
```

```
    },
    "requestID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE22222",
    "eventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLE33333",
    "readOnly": false,
    "resources": [
        {
            "accountId": "AWS Internal",
            "type": "AWS::KMS::Key",
            "ARN": "arn:aws:kms:us-west-2:account-id:key/a1b2c3d4-5678-90ab-cdef-
EXAMPLE11111"
        }
    ],
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "account-id",
    "sharedEventID": "a1b2c3d4-5678-90ab-cdef-EXAMPLEaaaaa",
    "eventCategory": "Management"
}
```

Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS

AWS est responsable de la maintenance de l'infrastructure sous-jacente pour AWS Fargate. AWS détermine quand une révision de version de plate-forme doit être remplacée par une nouvelle révision de l'infrastructure. C'est ce que l'on appelle le retrait des tâches. AWS envoie une notification de retrait de tâche lorsqu'une révision de version de plate-forme est retirée. Nous mettons régulièrement à jour les versions de nos plateformes prises en charge pour introduire une nouvelle révision contenant des mises à jour du logiciel d'exécution Fargate et des dépendances sous-jacentes telles que le système d'exploitation et l'environnement d'exécution du conteneur. Une fois qu'une nouvelle version est disponible, nous mettons hors service l'ancienne version afin de garantir que toutes les charges de travail des clients s'exécutent sur la version la plus récente de la version de plateforme Fargate. Lorsqu'une révision est retirée, toutes les tâches exécutées sur cette révision sont arrêtées.

Les tâches Amazon ECS peuvent être classées en tant que tâches de service ou tâches autonomes. Les tâches de service sont déployées dans le cadre d'un service et planifiées par le planificateur Amazon ECS. Pour plus d'informations, consultez Services Amazon ECS. Les tâches autonomes sont des tâches lancées par Amazon ECS `RunTask` API, soit directement, soit par un planificateur externe, tel que les tâches planifiées (qui sont lancées par Amazon) EventBridge), AWS Batch, ou

AWS Step Functions. Aucune action n'est requise de votre part en réponse à la mise hors service de vos tâches de service, car le planificateur Amazon ECS remplace automatiquement les tâches.

Pour les tâches autonomes, il se peut que vous deviez effectuer une manipulation supplémentaire en réponse à la mise hors service des tâches. Pour plus d'informations, consultez Amazon ECS peut-il gérer automatiquement des tâches autonomes ?.

Pour les tâches de service, vous n'avez aucune action à effectuer pour supprimer une tâche, sauf si vous souhaitez remplacer ces tâches au préalable. AWS fait. Lorsque le planificateur Amazon ECS arrête les tâches, il utilise le paramètre `maximumPercent` et lance une nouvelle tâche afin de maintenir le nombre souhaité pour le service. Suivez les pratiques exemplaires pour minimiser l'impact de la mise hors service des tâches. La valeur par défaut du paramètre `maximumPercent` pour un service utilisant le planificateur de service RÉPLICA est de 200 %. Par conséquent, lorsque AWS Fargate commence à supprimer des tâches, Amazon ECS planifie d'abord une nouvelle tâche, puis attend qu'elle soit exécutée, avant de supprimer une ancienne tâche. Lorsque vous définissez la valeur de `maximumPercent` sur 100 %, Amazon ECS arrête d'abord la tâche, puis la remplace.

Pour la suppression de tâches autonomes, AWS arrête la tâche à la date de retrait de la tâche ou après cette date. Amazon ECS ne lance pas de tâche de remplacement lorsqu'une tâche est arrêtée. Si vous souhaitez que ces tâches continuent de s'exécuter, vous devez arrêter les tâches en cours d'exécution et lancer une tâche de remplacement avant l'heure indiquée dans la notification. Nous recommandons donc aux clients de surveiller l'état des tâches autonomes et, si nécessaire, d'implémenter une logique pour remplacer les tâches interrompues.

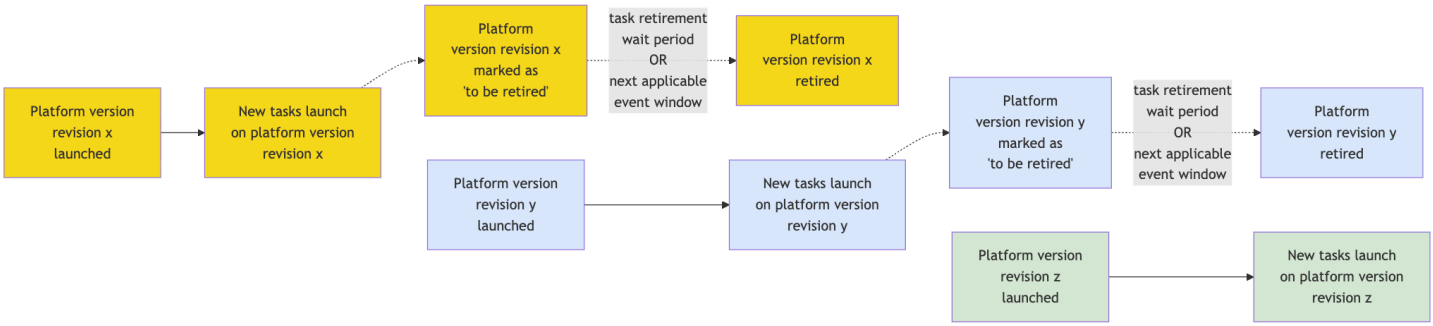
Lorsqu'une tâche est arrêtée dans l'un des scénarios ci-dessus, vous pouvez exécuter `describe-tasks`. La `stoppedReason` de la réponse est `ECS is performing maintenance on the underlying infrastructure hosting the task`.

La maintenance des tâches s'applique lorsqu'une nouvelle version de plateforme doit être remplacée par une nouvelle révision. En cas de problème avec un hôte Fargate sous-jacent, Amazon ECS remplace l'hôte sans préavis de mise hors service de tâche.

Aperçu des avis de mise hors service de tâches

Quand AWS marque une révision de version de plate-forme comme devant être supprimée, nous identifions toutes les tâches qui s'exécutent sur cette révision de version de plate-forme dans toutes les régions.

L'illustration suivante montre le cycle de vie d'une révision de version de plateforme Fargate, depuis le lancement d'une nouvelle révision à la mise hors service de la révision de plateforme.



Les informations suivantes fournissent des détails.

- Après le lancement d'une nouvelle révision de la version de plateforme, toutes les nouvelles tâches sont planifiées sur cette révision.
- Les tâches existantes qui ont été planifiées et exécutées restent dans la version dans laquelle elles ont été initialement placées pendant toute la durée de la tâche et ne sont pas migrées vers la nouvelle révision.
- Les nouvelles tâches, par exemple dans le cadre d'une mise à jour d'un service ou de la mise hors service d'une tâche Fargate, sont placées sur la dernière version de plateforme disponible au moment du lancement.

Les notifications de retrait des tâches sont envoyées via AWS Health Tableau de bord ainsi que par e-mail à l'adresse e-mail enregistrée et comprend les informations suivantes :

- Date de retrait de la tâche : la tâche est arrêtée à cette date ou après.
- Pour les tâches autonomes, les ID des tâches.
- Pour les tâches de service, l'ID du cluster sur lequel le service s'exécute et les ID du service.
- Les prochaines étapes que vous devez suivre.

En règle générale, nous envoyons une notification pour chaque tâche de service et chaque tâche autonome dans chaque Région AWS. Toutefois, dans certains cas, vous pouvez recevoir plusieurs événements pour chaque type de tâche, par exemple lorsqu'un trop grand nombre de tâches dépassent les limites fixées par nos mécanismes de notification pour être mises hors service.

Vous pouvez identifier les tâches planifiées pour le retrait en procédant ainsi :

- Le Tableau de bord Health

  AWS Health les notifications peuvent être envoyées via Amazon EventBridge pour un stockage d'archives tel qu'Amazon Simple Storage Service, effectuez des actions automatisées telles que l'exécution d'un AWS Lambda fonction, ou d'autres systèmes de notification tels qu'Amazon Simple Notification Service. Pour plus d'informations, voir Surveillance AWS Health événements avec Amazon EventBridge. Pour un exemple de configuration permettant d'envoyer des notifications à Amazon Chime, Slack ou Microsoft Teams, consultez AWS Health Conscient référentiel sur GitHub.

  Ce qui suit est un exemple EventBridge événement.

```
{
    "version": "0",
    "id": "3c268027-f43c-0171-7425-1d799EXAMPLE",
    "detail-type": "AWS Health Event",
    "source": "aws.health",
    "account": "123456789012",
    "time": "2023-08-16T23:18:51Z",
    "region": "us-east-1",
    "resources": [
        "cluster|service",
        "cluster|service"
    ],
    "detail": {
        "eventArn": "arn:aws:health:us-east-1::event/ECS/
AWS_ECS_TASK_PATCHING_RETIREMENT/AWS_ECS_TASK_PATCHING_RETIREMENT_test1",
        "service": "ECS",
```

```
        "eventScopeCode": "ACCOUNT_SPECIFIC",
        "communicationId":
 "7988399e2e6fb0b905ddc88e0e2de1fd17e4c9fa60349577446d95a18EXAMPLE",
        "lastUpdatedTime": "Wed, 16 Aug 2023 23:18:52 GMT",
        "eventRegion": "us-east-1",
        "eventTypeCode": "AWS_ECS_TASK_PATCHING_RETIREMENT",
        "eventTypeCategory": "scheduledChange",
        "startTime": "Wed, 16 Aug 2023 23:18:51 GMT",
        "endTime": "Fri, 18 Aug 2023 23:18:51 GMT",
        "eventDescription": [
            {
                "language": "en_US",
                "latestDescription": "\\nA software update has been deployed to
 Fargate which includes CVE patches or other critical patches. No action is required
 on your part. All new tasks launched automatically uses the latest software
 version. For existing tasks, your tasks need to be restarted in order for these
 updates to apply. Your tasks running as part of the following ECS Services will
 be automatically updated beginning Wed, 16 Aug 2023 23:18:51 GMT.\\n\\nAfter Wed,
 16 Aug 2023 23:18:51 GMT, the ECS scheduler will gradually replace these tasks,
 respecting the deployment settings for your service. Typically, services should
 see little to no interruption during the update and no action is required. When AWS
 stops tasks, AWS uses the minimum healthy percent (1) and launches a new task in
 an attempt to maintain the desired count for the service. By default, the minimum
 healthy percent of a service is 100 percent, so a new task is started first before
 a task is stopped. Service tasks are routinely replaced in the same way when
 you scale the service or deploy configuration changes or deploy task definition
 revisions. If you would like to control the timing of this restart you can update
 the service before Wed, 16 Aug 2023 23:18:51 GMT, by running the update-service
 command from the ECS command-line interface specifying force-new-deployment for
 services using Rolling update deployment type. For example:\\n\\n$ aws ecs update-
service -service service_name \\\n--cluster cluster_name -force-new-deployment\
\n\\nFor services using Blue/Green deployment type with AWS CodeDeploy:\\nPlease
 refer to create-deployment document (2) and create new deployment using same task
 definition revision.\\n\\nFor further details on ECS deployment types, please
 refer to ECS Deployment Developer Guide (1).\\nFor further details on Fargate's
 update process, please refer to the AWS Fargate User Guide (3).\\nIf you have
 any questions or concerns, please contact AWS Support (4).\\n\\n(1) https://
docs.aws.amazon.com/AmazonECS/latest/developerguide/deployment-types.html\\n(2)
 https://docs.aws.amazon.com/cli/latest/reference/deploy/create-deployment.html\\n(3)
 https://docs.aws.amazon.com/AmazonECS/latest/userguide/task-maintenance.html\\n(4)
 https://aws.amazon.com/support\\n\\nA list of your affected resources(s) can be
 found in the 'Affected resources' tab in the 'Cluster/ Service' format in the AWS
 Health Dashboard. \\n\\n"
            }
```

```
        ],
      "affectedEntities": [
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c23333"
                },
                {
                    "entityValue": "arn:aws:ecs:eu-west-1:111222333444:task/
examplecluster/00805ce1d81940b5a37398e5a2c25555"
                }
            }
        ]
    }
}
```

- E-mail

  Un e-mail est envoyé à l'adresse e-mail enregistrée pour le Compte AWS IDENTIFIANT.

Pour plus d’informations sur la manière de préparer la mise hors service d’une tâche, consultez la sectionPréparez-vous pour AWS Retrait des tâches Fargate sur Amazon ECS.

État de la ressource affectée dans un avis de retrait de tâche

Chaque ressource affectée répertoriée dans un avis de retrait de tâche possède un statut qui indique si cette tâche attend toujours d'être supprimée. Lorsque la suppression d'une tâche est en attente, son statut est Altéré. Cela indique que la tâche est toujours en cours d'exécution sur la révision de la version de la plateforme dont la mise hors service est prévue.

Lorsqu'une tâche cesse de s'exécuter, son statut passe à Résolu automatiquement, généralement dans les 24 heures. Cela s'applique si AWS vous avez retiré la tâche dans le cadre de la maintenance des tâches, ou vous l'avez arrêtée ou remplacée vous-même avant la mise hors service prévue. Une fois que toutes les ressources concernées dans un avis sont Résolu, AWS ferme l'avis de retraite.

Puis-je refuser la mise hors service d’une tâche ?

Non. Dans le cadre du AWS modèle de responsabilité partagée, AWS est responsable de la gestion et de la maintenance de l'infrastructure sous-jacente pour AWS Fargate. Cela inclut l’exécution de mises à jour périodiques de la plateforme pour garantir la sécurité et la stabilité. Ces mises à jour

sont automatiquement appliquées par AWS et ne sont pas une option que les clients peuvent refuser. C'est l'un des principaux avantages de l'utilisation d'un AWS Fargate par rapport à l'exécution de vos charges de travail sur des instances EC2, la responsabilité de la maintenance de la plate-forme sous-jacente est assurée par AWS. Ce modèle vous permet de vous concentrer sur vos applications plutôt que sur la maintenance de l'infrastructure. En appliquant automatiquement ces mises à jour de la plateforme, AWS est en mesure de maintenir l'environnement Fargate à jour et sécurisé, sans qu'aucune action ne soit requise de votre part en tant que client. Cela permet de fournir un environnement conteneurisé fiable et sécurisé pour exécuter vos charges de travail sur Fargate.

Puis-je recevoir des notifications de suppression de tâches par le biais d'autres AWS des services ?

AWS envoie une notification de retrait de tâche au Tableau de bord Health et au contact e-mail principal sur le Compte AWS. Le Tableau de bord Health fournit un certain nombre d'intégrations dans d'autres AWS services, y compris EventBridge. Vous pouvez utiliser EventBridge pour automatiser la visibilité des notifications (par exemple, transférer le message vers un ChatOps outil). Pour plus d'informations, consultez la section Présentation de la solution : capture des notifications de mise hors service de tâches.

Puis-je modifier la mise hors service d'une tâche une fois qu'elle a été planifiée ?

Non. Le calendrier est basé sur le temps d'attente de retrait des tâches, qui est par défaut de 7 jours. Si vous avez besoin de plus de temps, vous pouvez choisir de configurer le délai d'attente à 14 jours. Pour plus d'informations, consultez Étape 2 : capturer les notifications de mise hors service de tâche pour alerter les équipes et prendre des mesures.

À partir du 12/18/2025, Amazon ECS vous permet de configurer Fenêtres d'événements Amazon EC2 pour vos tâches Fargate. Si vous avez besoin d'un contrôle précis sur le calendrier exact des abandons de tâches, par exemple en les planifiant le week-end pour éviter toute interruption pendant les heures de bureau, vous pouvez configurer des fenêtres d'événements Amazon EC2 pour vos tâches, services ou clusters, voir Étape 1 : définissez le temps d'attente de la tâche ou utilisez les fenêtres d'événements Amazon EC2. Notez que la modification de cette configuration s'applique aux départs à la retraite qui seront planifiés dans le futur. Les mises hors service prévues actuellement ne sont pas concernées. En outre, lorsque vous configurez une fenêtre d'événements Amazon EC2 pour vos tâches Fargate, elle a priorité sur la configuration du temps d'attente de retrait de vos tâches. Si vous avez d'autres préoccupations, contactez Support.

Comment Amazon ECS gère-t-il les tâches faisant partie d'un service ?

Pour les tâches de service, vous n'avez aucune action à entreprendre en réponse à la suppression d'une tâche, sauf si vous souhaitez remplacer ces tâches avant AWS fait. Lorsque le planificateur Amazon ECS arrête les tâches, il utilise le pourcentage minimum sain et lance une nouvelle tâche afin de maintenir le nombre souhaité pour le service. Afin de minimiser l'impact de la mise hors service des tâches Fargate, les charges de travail doivent être déployées conformément aux pratiques exemplaires Amazon ECS. Par exemple, lors du déploiement d'une application sans état en tant que service Amazon ECS, tel qu'un serveur Web ou API, les clients doivent déployer plusieurs répliques de tâches et définir le minimumHealthyPercent jusqu'à 100 %. La valeur par défaut pour le pourcentage minimum sain est 100 %. Par conséquent, lorsque Fargate commence à mettre des tâches hors service, Amazon ECS planifie d'abord une nouvelle tâche et attend qu'elle soit exécutée, avant de mettre une ancienne tâche hors service. Les tâches de service sont systématiquement remplacées dans le cadre de la mise hors service des tâches, de la même manière que lorsque vous adaptez le service, déployez des modifications de configuration ou déployez des révisions de définition de tâches. Pour plus d'informations sur la mise hors service d'une définition de tâche, consultez la section Préparez-vous pour AWS Retrait des tâches Fargate sur Amazon ECS.

Amazon ECS peut-il gérer automatiquement des tâches autonomes ?

Non. AWS impossible de créer une tâche de remplacement pour les tâches autonomes démarrées par `RunTask`, des tâches planifiées (par exemple via EventBridge Planificateur), AWS Batch, ou AWS Step Functions. Amazon ECS ne gère que les tâches faisant partie d'un service.

Résolution des problèmes de disponibilité du service pendant la mise hors service d'une tâche

Si Amazon ECS ne peut pas démarrer une tâche de remplacement pendant la mise hors service d'une tâche, la disponibilité de votre service peut être affectée. Cela peut être dû à une configuration client incorrecte, telle que :

- Rôles IAM manquants ou configurés de façon incorrecte
- Capacité insuffisante dans les sous-réseaux cibles
- Mauvaises configurations des groupes de sécurité
- Erreurs de définition des tâches

Lorsqu'Amazon ECS ne peut pas lancer de tâches de remplacement, les tâches mises hors service sont arrêtées sans remplacement, ce qui réduit la capacité disponible de votre service et peut entraîner une interruption du service. Surveillez le nombre de tâches de votre service et Amazon CloudWatch des indicateurs pour garantir que les tâches de remplacement sont lancées avec succès lors des événements de retraite.

Préparez-vous pour AWS Retrait des tâches Fargate sur Amazon ECS

Afin d'être prêt à la mise hors service de tâche, effectuez les opérations suivantes :

1. Définissez la période d'attente de retrait des tâches ou utilisez les fenêtres d'événements Amazon EC2.
2. Capturez les notifications de mise hors service de tâche pour avertir les membres de l'équipe.
3. Vous pouvez vous assurer que toutes les tâches de vos services s'exécutent sur la dernière version de la plateforme en mettant à jour le service avec l'option de déploiement forcé. Cette étape est facultative.

Étape 1 : définissez le temps d'attente de la tâche ou utilisez les fenêtres d'événements Amazon EC2

Vous disposez de deux options de configuration de compte pour configurer l'heure à laquelle Fargate commence les abandons de tâches : `fargateTaskRetirementWaitPeriod` et `fargateEventWindows`.

Utilisation de fargateTaskRetirementWaitPeriod paramétrage du compte

Vous pouvez configurer l'heure à laquelle Fargate commence le retrait des tâches. Le délai d'attente par défaut est de 7 jours. Pour les charges de travail qui nécessitent l'application immédiate des mises à jour, choisissez le paramètre immédiat (`0`). Si vous avez besoin de plus de temps, configurez `7`, ou `14` option de jour.

Nous vous recommandons de choisir une période d'attente plus courte afin de pouvoir accéder plus rapidement aux nouvelles versions de plateforme.

Configurez la période d'attente en exécutant `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur racine ou utilisateur administratif. Utilisez l'option `fargateTaskRetirementWaitPeriod` pour le `name` et l'option `value` définie sur l'une des valeurs suivantes :

- `0` - AWS envoie la notification et commence immédiatement à supprimer les tâches concernées.
- `7` - AWS envoie la notification et attend 7 jours calendaires avant de commencer à retirer les tâches concernées. Il s'agit de l'option par défaut.
- `14` : AWS envoie la notification et attend 14 jours calendaires avant de commencer à retirer les tâches concernées.

Pour plus d'informations, veuillez consulter put-account-setting-default et put-account-setting dans la Référence de l'API Amazon Elastic Container Service.

Utilisation de fargateEventWindows paramétrage du compte

À partir du 12/18/2025, Amazon ECS vous permet de configurer Fenêtres d'événements Amazon EC2 pour vos tâches Fargate. Si vous avez besoin d'un contrôle précis sur le calendrier exact des abandons de tâches, par exemple en les planifiant le week-end pour éviter toute interruption pendant les heures de bureau, vous pouvez configurer des fenêtres d'événements Amazon EC2 pour vos tâches, services ou clusters.

Lorsque vous utilisez des fenêtres d'événements, Fargate garantit que vos tâches s'exécutent pendant au moins 3 jours avant d'être supprimées dans la prochaine fenêtre disponible, à moins qu'elles ne soient interrompues par des actions initiées par l'utilisateur ou par des problèmes de santé critiques tels que la dégradation du matériel sous-jacent.

Réglez le paramètre du compte `fargateEventWindows` sur `enabled`. Vous pouvez utiliser l'une des API suivantes : `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur root ou administrateur.

Chaque fenêtre d'événement Amazon EC2 doit être ouverte au moins 4 heures par semaine, et chaque plage horaire doit être d'au moins 2 heures. Pour les clusters et les services de grande taille, nous vous recommandons de configurer des fenêtres d'événements de longue durée (8 heures ou plus) ou de plages horaires plus fréquentes qui se produisent au moins une fois tous les 3 jours. Vous pouvez en savoir plus sur les considérations relatives aux fenêtres d'événements Amazon EC2 dans le guide de l'utilisateur AWS Fargate garantit que vos tâches s'exécutent pendant au moins 3 jours avant d'être supprimées, à moins qu'elles ne soient interrompues par des actions initiées par l'utilisateur ou par des problèmes de santé critiques tels qu'une dégradation du matériel sous-jacent.

> **Important**
>
> Le remplacement des tâches dans la fenêtre de l'événement est le meilleur effort. Si vous remarquez que des tâches sont supprimées en dehors des fenêtres de votre événement,

envisagez d'en prolonger la durée (8 heures ou plus) ou d'augmenter la fréquence (au moins une fois tous les 3 jours).

Pour appliquer les fenêtres d'événements Amazon EC2 à la suppression de vos tâches Fargate, procédez comme suit :

- Réglez le paramètre du compte `fargateEventWindows` sur `enabled`. Vous pouvez utiliser l'une des API suivantes : `put-account-setting-default` ou `put-account-setting` en tant qu'utilisateur root ou administrateur. Notez qu'il s'agit d'une activation unique pour l'utilisation de la fonctionnalité de fenêtres d'événements Amazon EC2 pour vos tâches Fargate.
- Créez une fenêtre d'événements Amazon EC2 via la console AWS ou l'AWS CLI. Pour créer une fenêtre d'événements à l'aide de la CLI, utilisez EC2 `create-instance-event-window` API avec plages temporelles ou expressions cron. Prenez note des `InstanceEventWindowId` à partir de la réponse.

  ```
  aws ec2 create-instance-event-window \
                      --time-range
   StartWeekDay=monday,StartHour=2,EndWeekDay=wednesday,EndHour=8 \
                      --tag-specifications "ResourceType=instance-event-
  window,Tags=[{Key=K1,Value=V1" \
                      --name myEventWindowName
  ```

  Vous pouvez également utiliser des expressions cron lors de la création de fenêtres d'événements EC2.

  ```
  aws ec2 create-instance-event-window \
                      --cron-expression "* 21-23 * * 2,3" \
                      --tag-specifications "ResourceType=instance-event-
  window,Tags=[{Key=K1,Value=V1" \
                      --name myEventWindowName
  ```

- Vous pouvez ensuite associer la fenêtre de l'événement à des services spécifiques, à des clusters ou à toutes les tâches de votre compte à l'aide d'EC2 `associate-instance-event-window` API.
  - Pour les tâches de service ECS

    ```
    aws ec2 associate-instance-event-window \
    --instance-event-window-id iew-0abcdef1234567890 \
    ```

```
--association-target "InstanceTags=[{Key=aws:ecs:serviceArn,Value=your-service-
arn}]"
```

- Pour les clusters ECS

```
aws ec2 associate-instance-event-window \
--instance-event-window-id iew-0abcdef1234567890 \
--association-target "InstanceTags=[{Key=aws:ecs:clusterArn,Value=your-cluster-
arn}]"
```

- Pour associer une fenêtre d'événements à toutes les tâches du compte

```
aws ec2 associate-instance-event-window \
--instance-event-window-id iew-0abcdef1234567890 \
--association-target "InstanceTags=[{Key=aws:ecs:fargateTask,Value=true}]"
```

Vous pouvez utiliser plusieurs paires clé-valeur pour associer une fenêtre d'événement à plusieurs services ou clusters.

Fargate choisira la fenêtre d'événement pour chaque tâche dans l'ordre suivant :

- Si une fenêtre d'événement est associée au service de la tâche, elle sera utilisée. Cela ne s'applique pas aux tâches autonomes ou non gérées.
- Si une fenêtre d'événements est associée au cluster de la tâche, elle sera utilisée.
- Si une fenêtre d'événements est définie pour toutes les tâches de Fargate, elle sera utilisée.
- Le `fargateTaskRetirementWaitPeriod` le paramètre sera utilisé si aucune des fenêtres d'événements ne correspond à la tâche.

Configuration des fenêtres d'événements pour la maintenance des tâches Fargate

Prenons le cas où vous exécutez plusieurs services ECS sur Fargate avec des exigences de disponibilité différentes. Vous souhaitez un contrôle précis des abandons de tâches. Vous pouvez configurer plusieurs fenêtres d'événements comme suit :

- Maintenance par défaut pour toutes les tâches Fargate: créez une fenêtre d'événements pour la maintenance de routine pendant les heures creuses (de midi à 4 h du matin tous les jours) et associez-la à toutes les tâches de Fargate à l'aide du `aws:ecs:fargateTask` étiquette.
- Weekend-only cluster de maintenance pour le développement: pour un cluster de développement dont les services peuvent tolérer des interruptions le week-end, créez une fenêtre de 24 heures sur 24 le week-end (samedi et dimanche, toute la journée) et associez-la au cluster à l'aide du `aws:ecs:clusterArn` étiquette avec l'ARN de votre cluster.
- Fenêtre restreinte pour les services critiques: pour un service de traitement des paiements essentiel nécessitant une disponibilité élevée en semaine, limitez la maintenance aux petites heures du week-end (samedi et dimanche, de midi à 4 h) et associez-la au service spécifique à l'aide du `aws:ecs:serviceArn` étiquette avec l'ARN de votre service.

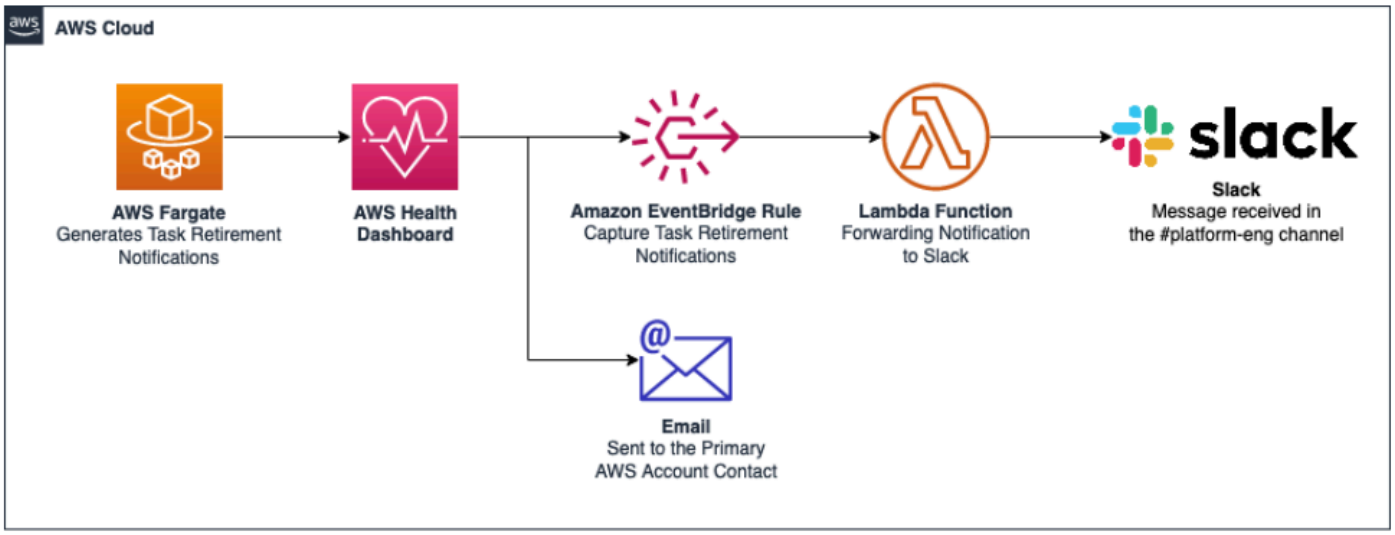
Avec cette configuration, le service de paiement utilise sa fenêtre spécifique uniquement le week-end, les services et tâches du cluster de développement utilisent la fenêtre de 24 heures du week-end et toutes les autres tâches Fargate utilisent la fenêtre de maintenance quotidienne par défaut.

Pour plus d'informations, veuillez consulter put-account-setting-default et put-account-setting dans la Référence de l'API Amazon Elastic Container Service.

Étape 2 : capturer les notifications de mise hors service de tâche pour alerter les équipes et prendre des mesures

En cas de retrait d'une tâche à venir, AWS envoie une notification de retrait de tâche au AWS Health Tableau de bord et au contact e-mail principal sur le Compte AWS. Le AWS Health Le tableau de bord propose un certain nombre d'intégrations dans d'autres AWS services, y compris Amazon EventBridge. Vous pouvez utiliser EventBridge pour créer des automatisations à partir d'une notification de retrait de tâche, par exemple en augmentant la visibilité du retrait à venir en transférant le message à un ChatOps outil. AWS Health Aware est une ressource qui montre la puissance du AWS Health Tableau de bord et comment les notifications peuvent être distribuées au sein d'une organisation. Vous pouvez transférer une notification de mise hors service de tâche vers une application de chat, telle que Slack.

L’illustration suivante présente une vue d’ensemble de la solution.



Les informations suivantes fournissent des détails.

- Fargate envoie la notification de mise hors service de la tâche au tableau de bord AWS Health .
- Le AWS Health Le tableau de bord envoie un e-mail au contact e-mail principal sur le Compte AWS, et notifie EventBridge.
- EventBridge dispose d'une règle qui capture la notification de retrait.

  La règle qui recherche les événements avec le type de détail de l'événement : `"AWS Health Event" and the Event Detail Type Code: "AWS_ECS_TASK_PATCHING_RETIREMENT"`
- La règle déclenche une fonction Lambda qui transmet les informations à Slack à l'aide d'un Webhook entrant Slack. Pour plus d'informations, consultez la section Webhooks entrants.

Pour un exemple de code, voir Capture AWS Fargate Notifications de retrait de tâches sur Github.

Étape 3 : contrôler le remplacement des tâches

Vous ne pouvez pas contrôler le moment exact de la mise hors service d'une tâche, mais vous pouvez définir un temps d'attente. Si vous souhaitez contrôler le remplacement des tâches selon votre propre calendrier, vous pouvez enregistrer l'avis de mise hors service d'une tâche afin de comprendre d'abord la date de mise hors service de la tâche. Vous pouvez ensuite redéployer votre service pour lancer des tâches de remplacement, ainsi que pour remplacer toutes les tâches autonomes. Pour les services utilisant un déploiement continu, vous devez mettre à jour le service

à l'aide de `update-service` avec l'option `force-deployment` avant l'heure de début de la mise hors service.

L'exemple suivant `update-service` utilise l'option `force-deployment`.

```
aws ecs update-service —-service service_name \
    --cluster cluster_name \
     --force-new-deployment
```

Pour les services qui utilisent le blue/green déploiement, vous devez créer un nouveau déploiement dans AWS CodeDeploy. Pour plus d'informations sur la façon de créer le déploiement, consultez la section create-deployment dans la Référence AWS Command Line Interface .

Régions prises en charge pour Amazon ECS sur AWS Fargate

Vous pouvez utiliser les tableaux suivants pour vérifier la prise en charge régionale des conteneurs Linux sur AWS Conteneurs Fargate et Windows sur AWS Fargate.

Conteneurs Linux sur AWS Fargate

Conteneurs Amazon ECS Linux sur AWS Fargate sont pris en charge dans les domaines suivants Régions AWS. Les ID de zone de disponibilité pris en charge sont indiqués le cas échéant.

Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1
USA Ouest (Californie du Nord)	us-west-1 (`usw1-az1` et `usw1-az3` uniquement)
USA Ouest (Oregon)	us-west-2
Canada (Centre)	ca-central-1
Canada-Ouest (Calgary)	ca-west-1
Mexique (Centre)	mx-central-1

Nom de la région	Région
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Tokyo)	ap-northeast-1 (`apne1-az1` , `apne1-az2` et `apne1-az4` uniquement)
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Thaïlande)	ap-southeast-7
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Malaisie)	ap-southeast-5
Canada (Centre)	ca-central-1
Canada-Ouest (Calgary)	ca-west-1
Chine (Pékin)	cn-north-1 (`cnn1-az1` et `cnn1-az2` uniquement)
Chine (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2

Nom de la région	Région
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Espagne)	eu-south-2
Europe (Stockholm)	eu-north-1
Amérique du Sud (São Paulo)	sa-east-1
Israël (Tel Aviv)	il-central-1
Moyen-Orient (Bahreïn)	me-south-1
Moyen-Orient (EAU)	me-central-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

Conteneurs Windows allumés AWS Fargate

Conteneurs Windows Amazon ECS sur AWS Fargate sont pris en charge dans les domaines suivants Régions AWS. Les ID de zone de disponibilité pris en charge sont indiqués le cas échéant.

Nom de la région	Région
USA Est (Ohio)	us-east-2
USA Est (Virginie du Nord)	us-east-1 (`use1-az1`, `use1-az2`, `use1-az4`, `use1-az5` et `use1-az6` uniquement)
USA Ouest (Californie du Nord)	us-west-1 (`usw1-az1` et `usw1-az3` uniquement)

Nom de la région	Région
USA Ouest (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Malaisie)	ap-southeast-5
Asie-Pacifique (Tokyo)	ap-northeast-1 (`apne1-az1` , `apne1-az2` et `apne1-az4` uniquement)
Canada (Centre)	ca-central-1 (`cac1-az1` et `cac1-az2` uniquement)
Canada-Ouest (Calgary)	ca-west-1
Chine (Pékin)	cn-north-1 (`cnn1-az1` et `cnn1-az2` uniquement)
Chine (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Irlande)	eu-west-1

Nom de la région	Région
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Espagne)	eu-south-2
Europe (Stockholm)	eu-north-1
Amérique du Sud (São Paulo)	sa-east-1
Israël (Tel Aviv)	il-central-1
Moyen-Orient (EAU)	me-central-1
Moyen-Orient (Bahreïn)	me-south-1

Architecture de la capacité EC2 pour Amazon ECS

Utilisez la capacité EC2 pour les charges de travail volumineuses dont le prix doit être optimisé.

Lorsque vous envisagez de modéliser les définitions de tâches et les services en utilisant EC2, nous vous recommandons de déterminer quels processus doivent s'exécuter ensemble et comment vous pouvez procéder pour mettre à l'échelle chaque composant.

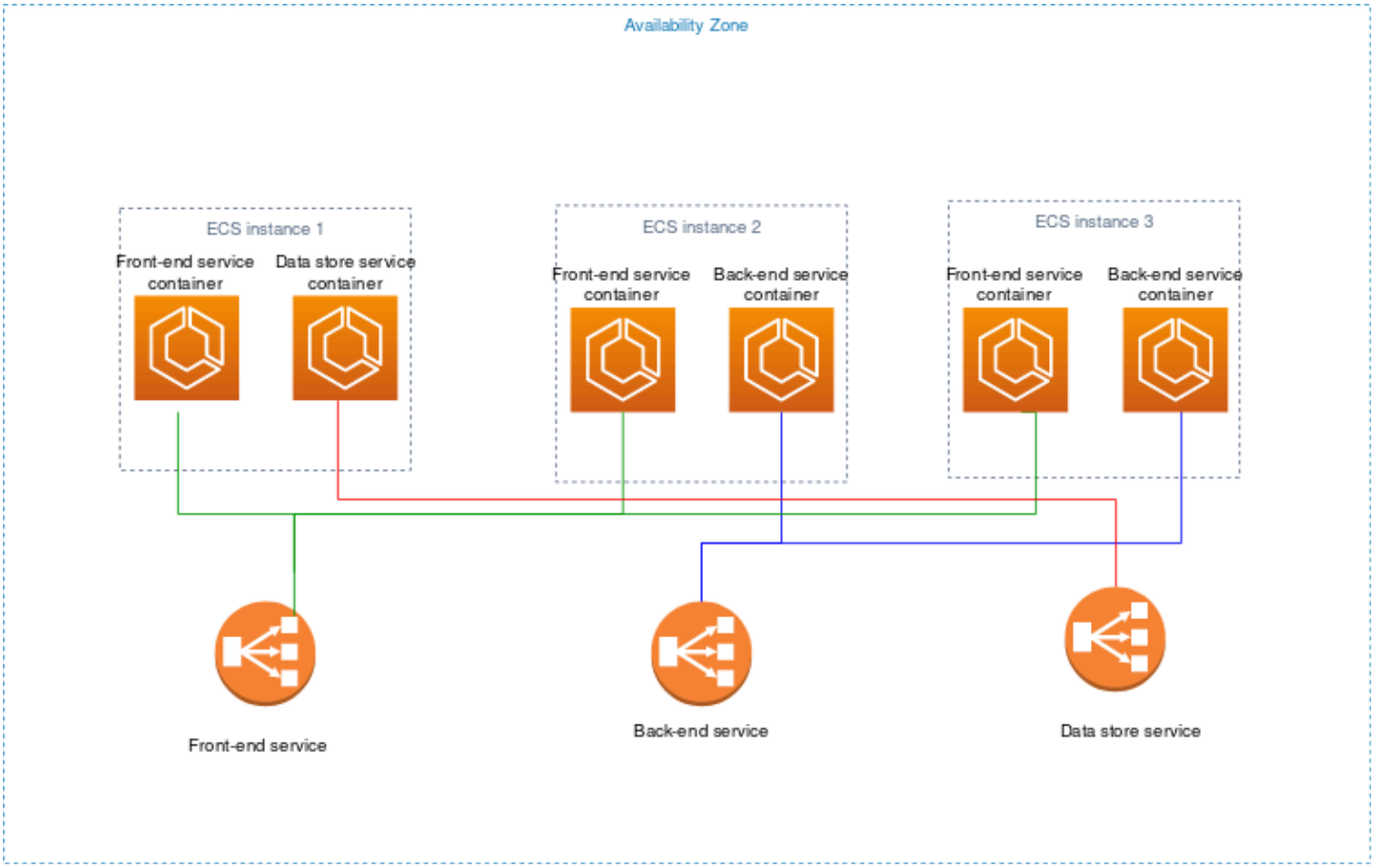
Par exemple, supposons qu'une application comporte les composants suivants :

- Un service frontal qui affiche des informations sur une page web
- Un service backend qui fournit les API pour le service frontal
- Une banque de données

Pour cet exemple, créez des définitions de tâches qui regroupent les conteneurs utilisés dans un même but. Séparez les différents composants en plusieurs définitions de tâches. L'exemple de cluster suivant comporte trois instances de conteneur qui exécutent trois conteneurs de service frontal, deux conteneurs de service backend et un conteneur de services de magasin de données.

Vous pouvez regrouper les conteneurs connexes dans une définition de tâche, comme les conteneurs liés qui doivent être exécutés ensemble. Par exemple, ajoutez un conteneur de flux de journaux à votre service frontal et l'inclure dans la même définition de tâche.

Une fois que vous avez créé vos définitions de tâche, vous pouvez créer des services à partir de ces dernières afin de préserver la disponibilité des tâches souhaitées. Pour de plus amples informations, veuillez consulter Création d'un déploiement de mise à jour propagée Amazon ECS. Dans vos services, vous pouvez associer des conteneurs avec des équilibreurs de charge Elastic Load Balancing. Pour de plus amples informations, veuillez consulter Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS. Lorsque les besoins de votre application changent, vous pouvez mettre à jour vos services pour augmenter ou réduire le nombre de tâches souhaitées. Vous pouvez également mettre à jour vos services afin de déployer des versions plus récentes des conteneurs dans vos tâches. Pour de plus amples informations, veuillez consulter Mettre à jour un service Amazon ECS.



Comportement d'extraction d'image de conteneur pour EC2 et les instances externes sur Amazon ECS

Le temps nécessaire au démarrage d'un conteneur varie en fonction de l'image du conteneur sous-jacente. Par exemple, le démarrage d'une image plus volumineuse (versions complètes de Debian, Ubuntu et Amazon1/2) peut prendre plus de temps car les conteneurs contiennent un plus grand nombre de services que leurs versions allégées respectives (Debian-slim, Ubuntu-slim, et Amazon-slim) ou les images de base plus petites (Alpine).

Lorsque l'agent Amazon ECS lance une tâche, il extrait l'image Docker de son registre distant, puis met en cache une copie locale. Lorsque vous utilisez une nouvelle balise d'image pour chaque version de votre application, ce comportement n'est pas nécessaire.

Le paramètre de l'agent `ECS_IMAGE_PULL_BEHAVIOR` détermine le comportement d'extraction de l'image. Les options suivantes sont disponibles :

- `ECS_IMAGE_PULL_BEHAVIOR`: `default`

L’image sera extraite à distance. Si l’extraction échoue, l’image mise en cache dans l’instance est utilisée.

- `ECS_IMAGE_PULL_BEHAVIOR`: `always`

  L’image sera extraite à distance. Si l’extraction échoue, la tâche échoue.

Pour accélérer le déploiement, définissez le paramètre de l’agent Amazon ECS sur l’une des valeurs suivantes :

- `ECS_IMAGE_PULL_BEHAVIOR`: `once`

  L’image n’est extraite à distance que si elle n’a pas été extraite par une tâche précédente sur la même instance de conteneur ou si l’image mise en cache a été supprimée par le processus de nettoyage automatique des images. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Cela garantit qu'aucune extraction d'image inutile ne soit tentée.
- `ECS_IMAGE_PULL_BEHAVIOR`: `prefer-cached`

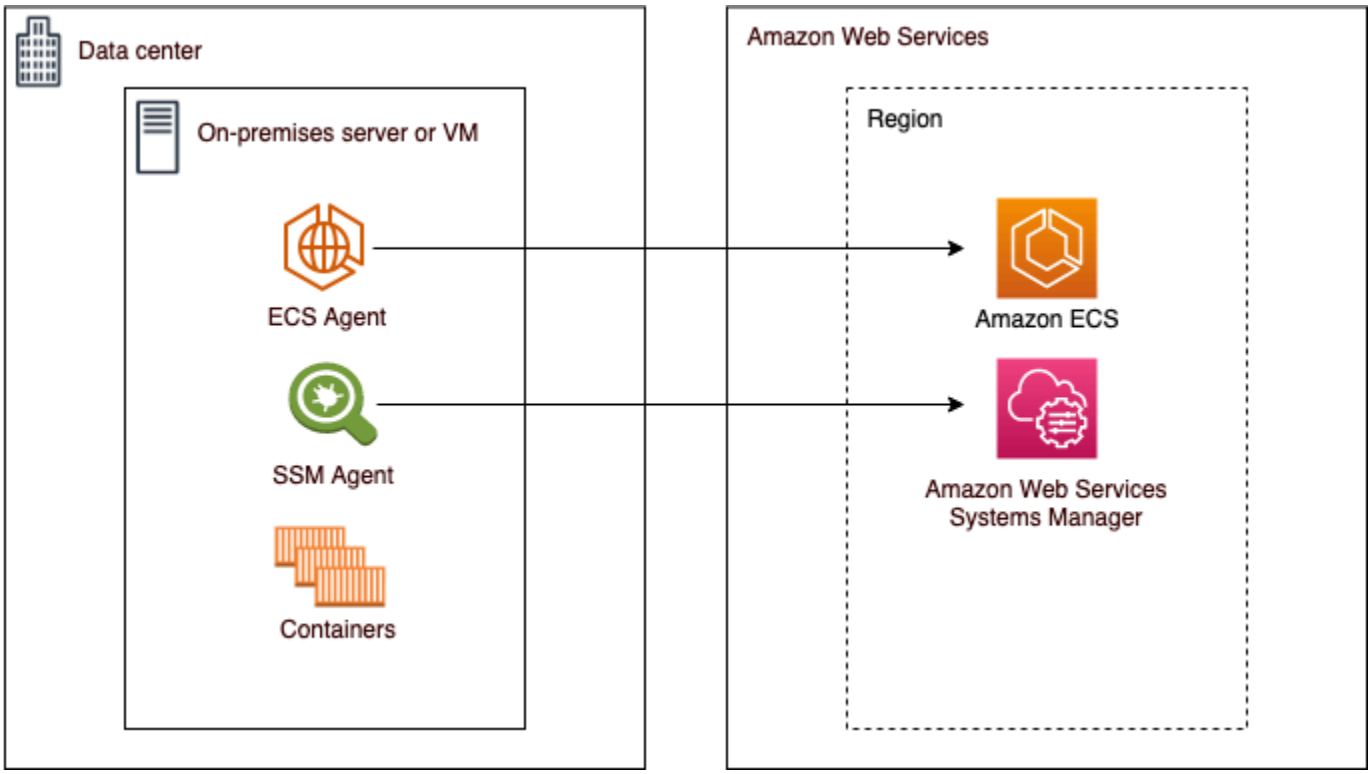
  L’image est extraite à distance s’il n’y a pas d’image mise en cache. Dans le cas contraire, l'image mise en cache sur l'instance est utilisée. Le nettoyage d’image automatique est désactivé pour le conteneur afin de garantir que l’image mise en cache ne soit pas supprimée.

La définition du paramètre `ECS_IMAGE_PULL_BEHAVIOR` sur l’une des valeurs précédentes permet de gagner du temps, car l’agent Amazon ECS utilise l’image téléchargée existante. Pour les images Docker de plus grande taille, le temps de téléchargement peut prendre de 10 à 20 secondes pour être transférées sur le réseau.

Amazon ECS Anywhere) externe pour Amazon ECS

Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS. Les instances externes sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de processus. Si votre application nécessite du trafic entrant, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace. Amazon ECS a ajouté un nouveau type de lancement `EXTERNAL` que vous pouvez utiliser pour créer des services ou exécuter des tâches sur vos instances externes.

Vous trouverez ci-dessous une présentation de l'architecture système de haut niveau d'Amazon ECS Anywhere. Votre serveur sur site dispose à la fois de l'agent Amazon ECS et de l'agent SSM.



Pour de plus amples informations, veuillez consulter Clusters Amazon ECS pour instances externes.

Mode Amazon ECS Express

Un service Amazon ECS Express Mode réduit la complexité du déploiement d'applications conteneurisées en fournissant des valeurs par défaut judicieuses et en automatisant la configuration du support AWS services. Au lieu de gérer les paramètres de configuration de plusieurs services, un service en mode Express ne nécessite que trois éléments pour démarrer :

- Une image de conteneur
- Un rôle d'exécution des tâches
- Un rôle d'infrastructure

Le mode Amazon ECS Express orchestre et configure toutes les infrastructures nécessaires : un Fargate-based Service ECS avec une URL accessible unique, équilibreur de charge avec SSL/TLS, politiques de dimensionnement automatique, surveillance et composants réseau.

Utilisez les services Amazon ECS Express Mode dans les scénarios suivants :

- Applications Web et API : applications conteneurisées sans état qui traitent des requêtes HTTP
- Prototypage rapide : déployez et testez rapidement des applications sans frais de configuration de l'infrastructure
- Productivité des développeurs : permettez aux équipes chargées des applications de déployer leurs applications de manière indépendante, sans nécessiter de AWS connaissances
- Efficacité de l'équipe chargée de la plateforme : réduisez les frais de maintenance en fournissant des fonctionnalités de déploiement en libre-service

Les services Amazon ECS Express Mode prennent en charge les applications HTTPS publiques ou privées et évoluent automatiquement en fonction de l'utilisation ou du trafic.

Avantages des services Amazon ECS Express Mode

Les services Amazon ECS Express Mode offrent plusieurs avantages clés pour le déploiement et la gestion d'applications conteneurisées sur AWS.

- Déploiement simplifié : déployez des applications conteneurisées avec des paramètres par défaut prêts à être utilisés en production sur plusieurs AWS services.

- Aucun compromis sur les capacités, toutes sous-jacentes AWS les ressources restent accessibles pour une gestion directe lorsque vous avez besoin d'un contrôle précis ou de fonctionnalités avancées.
- Optimisation des coûts : partage les équilibreurs de charge des applications entre plusieurs services en mode Express en utilisant la même configuration réseau afin de réduire les coûts.
- Infrastructure transparente - Toutes les ressources sont créées dans votre AWS compte avec une visibilité et une accessibilité complètes via le AWS console et API.

Tarification

L'utilisation d'un service Amazon ECS Express Mode est gratuite. Vous ne payez que pour le sous-jacent AWS les ressources créées pour exécuter votre application, notamment :

- Ressources informatiques Fargate
- Application Load Balancer
- CloudWatch journaux et statistiques
- Frais de transfert de données

Disponibilité

Le mode Amazon ECS Express est disponible dans tous AWS Régions dans lesquelles Amazon ECS et Fargate sont pris en charge. Vous pouvez créer et gérer des services en mode Express à l'aide de l'une des méthodes suivantes :

- La console Amazon ECS
- AWS AWS CLI
- AWS SDK
- CloudFormation
- Terraform
- AWS Serveur MCP Labs pour Amazon ECS

Création d'un service Amazon ECS Express Mode

Un service Amazon ECS Express Mode réduit la complexité du déploiement d'applications conteneurisées en fournissant des valeurs par défaut judicieuses et en automatisant la configuration du support AWS services. Au lieu de gérer les paramètres de configuration de plusieurs services, un service en mode Express ne nécessite qu'une image de conteneur, un rôle d'exécution des tâches et un rôle d'infrastructure pour démarrer.

Une fois votre déploiement terminé, une URL unique pour votre application vous sera renvoyée. Vous pouvez trouver le service que vous venez de déployer dans Clusters, et si vous n'avez pas spécifié de cluster, il figurera dans le `Default` cluster.

Conditions préalables

Avant de créer un service en mode Express, assurez-vous de disposer de l'un des éléments suivants :

- Une image de conteneur stockée dans Amazon ECR.
- Ou une image de conteneur stockée dans un registre privé. Pour utiliser un registre privé, configurez un secret Secrets Manager. Pour plus d'informations, voir Créez un AWS Secrets Manager secret dans le AWS Secrets Manager Guide de l'utilisateur.

> **Note**
>
> Les rôles IAM sont également obligatoires, mais ils sont abordés dans le Console et CLI guides séparément. La console Amazon ECS dispose de flux automatisés pour la création de rôles.

Procédures

- Pour une première utilisation simplifiée de la console, voir the section called “Créez votre premier service Express Mode dans la console”.
- Pour créer un service en mode Express à l'aide du AWS CLI, voir the section called “Créez votre premier service en mode Express à l'aide du AWS CLI”.
- Pour en savoir plus sur ce que crée le mode Express et sur son fonctionnement, voir the section called “Ressources créées par les services Express Mode ”.

Personnalisation

Lors de la création d'un service en mode Express, Amazon ECS sélectionne les valeurs par défaut pour votre Amazon ECS et d'autres AWS ressources. Personnalisez l'application si nécessaire à l'aide des paramètres facultatifs. Profitez d'un service en mode Express et personnalisez les éléments suivants :

- Nom du service
- Cluster Amazon ECS
- Paramètres du conteneur : port et chemin de contrôle de santé
- Variables d’environnement et secrets
- Commandes personnalisées
- Rôle de la tâche : ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.

  Vous avez besoin de ce rôle lorsque votre application accède à d'autres AWS des services, tels qu'Amazon S3.
- Ressources de calcul : allocation du processeur et de la mémoire
- Architecture du processeur : x86 ou ARM64 (AWS Calculer (Graviton)
- Configuration réseau : sous-réseaux et groupe de sécurité des services
- Stratégies d’autoscaling de l’application : métriques, valeurs cibles et limites des tâches
- CloudWatch Configuration des journaux
- Balises de ressources
- Définition de tâche : utilisez votre propre définition de tâche personnalisée

Créez votre premier service Amazon ECS Express Mode dans la console

L'expérience de console pour le service Express Mode fournit un moyen rationalisé de déployer votre application conteneurisée avec une configuration minimale requise. Pour en savoir plus sur les prérequis, consultez Mode Amazon ECS Express. Pour en savoir plus sur ce que crée le mode Express et sur son fonctionnement, voir the section called “Ressources créées par les services Express Mode ”.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le volet de navigation, choisissez Mode express.
3. En dessous Configurons votre application:

    a. Spécifiez l'image à utiliser pour votre application. Pour URI de l'image, entrez l'URI de votre image. Pour parcourir vos images Amazon ECR, choisissez Parcourez les images ECR, puis procédez comme suit :

        i. Pour Dépôt privé, choisissez le référentiel privé Amazon ECR.

        ii. Pour Image, choisissez votre image.

        iii. Choisissez le mode d'identification de l'image. Pour Sélectionnez l'image par, choisissez l'une des options suivantes :

            - AWS vous recommande de choisir Résumé d'images.
            - Pour utiliser la balise, choisissez Balise d'image puis choisissez le tag.

    b. Si vous utilisez un registre privé, sélectionnez Registre privé. Puis pour ARN ou nom de Secrets Manager, entrez l'ARN Secrets Manager que vous avez créé dans les prérequis.

    c. Pour Rôle d'exécution des tâches, et Rôle de l'infrastructure, choisissez les rôles. Si vous n'avez jamais créé ces rôles dans Amazon ECS auparavant, vous verrez l'option Créer un nouveau rôle dans la liste déroulante. Lorsque vous sélectionnez cette option, le rôle sera automatiquement créé pour vous une fois que vous aurez cliqué Créez. Pour le Rôle d'exécution des tâches, les détails de l'autorisation afficheront le lien pour plus de détails sur le AmazonECSTaskExecutionRolePolicy dans le AWS Guide de référence des politiques gérées. Pour le Rôle de l'infrastructure, les détails de l'autorisation afficheront le lien pour plus de détails sur le AmazonECSInfrastructureRoleforExpressGatewayServices Politique gérée dans AWS Guide de référence des politiques gérées.

        Si vous préférez personnaliser ces rôles, Créer un nouveau rôle Le bouton vous dirige vers la console IAM où vous pouvez créer le rôle, afficher et modifier les autorisations associées au rôle. Lorsque vous revenez à la console Amazon ECS Express Mode, actualisez pour afficher le rôle que vous avez créé.

> **Note**
>
> La première fois que vous créez un service dans la console Amazon ECS, certains utilisateurs peuvent voir un `Invalid Parameter Exception: Unable to assume the service linked role.` Dans ce cas, patientez quelques secondes et réessayez.

4. Cliquez sur Créer. Vous venez de créer votre premier service Amazon ECS Express Mode !

Étapes suivantes

Après avoir créé votre premier service en mode Express :

- Accédez à votre application à l'aide de l'URL d'application fournie une fois le déploiement terminé.
- Surveillez la progression du déploiement dans l'onglet Ressources de la console. Une fois le déploiement terminé, votre service est prêt à recevoir du trafic. Pour plus d'informations, consultez the section called “Afficher les détails d'un service en mode Express”.
- Pour revenir à votre service, sélectionnez dans le menu de navigation Clusters, puis sélectionnez `default` Cluster si vous n'en avez pas spécifié un lors de la création. Amazon ECS organise les services en groupes logiques appelés clusters. Il s'agit d'un moyen simple d'organiser les candidatures dans votre compte.
- Si vous rencontrez des problèmes lors du déploiement, consultez the section called “Résolution des problèmes”.

Personnalisation

En option, vous pouvez également personnaliser la configuration de votre service Express Mode. Ouvrez la section intitulée Configurations supplémentaires - optionnel.

1. Sélectionnez un Cluster dans la liste déroulante s'il existe un cluster Amazon ECS spécifique sur lequel vous souhaitez déployer votre service. Si aucun cluster n'est spécifié, le mode Express utilisera le `Default` grappe. Si vous quittez la console du mode Express pour créer un cluster, utilisez le bouton d'actualisation pour remplir la liste déroulante.
2. Entrez un Nom pour le service. Si aucun nom n'est spécifié, le mode Express en génère un à partir du nom de l'image du conteneur. Ce nom de service sera utilisé sur plusieurs ressources,

notamment l'URL de l'application, le service Amazon ECS, la définition des tâches Amazon ECS, le certificat, la cible de dimensionnement et la politique de dimensionnement.

3. Spécifiez éventuellement des informations sur votre Récipient:

   a. Pour Port à conteneurs, entrez le port sur lequel votre application écoute (la valeur par défaut est 80).

   b. Pour Parcours de contrôle de santé, entrez le chemin pour les bilans de santé (par exemple, `"/health"`). La valeur par défaut est `"/"`..

   c. En dessous Variables d'environnement, ajoutez des paires clé-valeur pour les variables d'environnement dont votre application a besoin. Pour Clé, entrez le nom de la variable d'environnement. Pour Type de valeur, choisissez Variable d'environnement ou Secret. Pour Valeur ou valeur provenant de, entrez la valeur ou la référence. Choisissez Ajouter une variable d'environnement pour ajouter d'autres variables si nécessaire. Aucune valeur par défaut n'est fournie.

   d. Pour Commande, entrez éventuellement une commande personnalisée pour remplacer le Docker `CMD` instruction. Aucune valeur par défaut n'est fournie.

   e. Pour Rôle de la tâche, choisissez un rôle IAM qui accorde des autorisations à votre application. Aucune valeur par défaut n'est fournie.

4. Spécifiez éventuellement des informations sur votre Calculer et Mise à l'échelle automatique:

   a. Pour CPU, choisissez l'allocation de processeur virtuel pour vos tâches (la valeur par défaut est de 1 processeur virtuel).

   b. Pour Mémoire, choisissez l'allocation de mémoire pour vos tâches (la valeur par défaut est de 2 Go).

   c. Pour Architecture du processeur, choisissez X86_64 ou BRAS 64 (la valeur par défaut est X86_64). Choisissez BRAS 64 pour exécuter vos tâches AWS Graviton-powered calculer.

   d. Pour Indicateur de service ECS, choisissez la métrique sur laquelle vous souhaitez effectuer l'échelle (la valeur par défaut est Utilisation moyenne du processeur).

   e. Pour Valeur cible, choisissez le pourcentage cible pour la mise à l'échelle (la valeur par défaut est 60).

   f. Pour Nombre minimum de tâches et Nombre maximum de tâches, définissez les limites de mise à l'échelle. (Les valeurs par défaut sont 1 et 20)

5. Cochez la case pour Personnalisez les configurations réseau. Si vous ne personnalisez pas ces configurations, Express Mode utilisera le VPC par défaut.

a. Sélectionnez un VPC pour vous aider à accéder rapidement au Sous-réseaux où vos services fonctionneront. Si vous le souhaitez, quittez la console Amazon ECS pour créer un nouveau VPC, puis revenez pour actualiser et rechercher ce VPC dans la liste déroulante.

b. Pour Groupes de sécurité, choisissez ou créez des groupes de sécurité pour autoriser un accès réseau entrant supplémentaire à votre service.

6. En dessous Journaux:

a. Pour Amazon CloudWatch groupe de journaux, entrez le nom de groupe de journaux préféré pour les journaux de votre application. Le groupe de journaux par défaut est nommé en fonction des noms de votre cluster et de votre service.

b. Pour Amazon CloudWatch préfixe du flux de journal, entrez un préfixe préféré pour les flux de journaux. Le préfixe de flux par défaut est `ecs/Main/.`

7. En dessous Balises, ajoutez des paires clé-valeur pour baliser vos ressources. Pour Key (Clé), saisissez la clé de l'identification. Pour Value (Valeur), saisissez la valeur de l'identification. Choisissez Ajouter un nouvel article pour ajouter d'autres balises si nécessaire. Les balises ne peuvent être ajoutées que lors de la création.

Créez votre premier service en mode Express à l'aide du AWS CLI

Ce didacticiel explique comment créer une application de service en mode Express à l'aide du AWS AWS CLI. Vous allez déployer une application Web simple qui présente les fonctionnalités de base du service Express Mode.

Conditions préalables

Avant de commencer, assurez-vous de disposer des éléments suivants :

- Un AWS compte avec les autorisations appropriées
- Le AWS AWS CLI installé et configuré
- Une image de conteneur stockée dans Amazon ECR ou dans un registre privé
- Un VPC par défaut avec des sous-réseaux publics dans le AWS Région dans laquelle vous créez le service. Par défaut, le service Express Mode crée un équilibreur de charge connecté à Internet dans le VPC et les sous-réseaux publics par défaut de votre compte. Si votre compte ne possède pas de VPC par défaut, créez-en un avant de commencer ou spécifiez vos propres sous-réseaux

lors de la création du service. Pour plus d'informations, voir VPC par défaut dans le Guide de l'utilisateur d'Amazon VPC.

Étape 1 : Création de rôles IAM

Un service en mode Express nécessite deux rôles IAM. Le rôle d'exécution des tâches permet à Amazon ECS d'extraire des images de conteneurs et de rédiger des journaux en votre nom. Créez un rôle d'exécution des tâches et un rôle d'infrastructure avec les politiques suivantes :

```
#Create the roles with ECS trust policies
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"
            }
        ]
    }'

aws iam create-role --role-name ecsInfrastructureRoleForExpressServices \
    --assume-role-policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Sid": "AllowAccessInfrastructureForECSExpressServices",
                "Effect": "Allow",
                "Principal": {
                    "Service": "ecs.amazonaws.com"
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'
```

```
#Attach the AWS managed policies
aws iam attach-role-policy --role-name ecsTaskExecutionRole \
```

```
    --policy-arn arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy
aws iam attach-role-policy --role-name ecsInfrastructureRoleForExpressServices \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRoleforExpressGatewayServices
```

Pour plus d'informations, consultez the section called "Rôle IAM d'exécution de tâche".

> **Note**
>
> Les rôles IAM sont finalement cohérents, de sorte qu'un rôle nouvellement créé peut ne pas être utilisable immédiatement. Si votre premier `create-express-gateway-service` l'appel échoue avec `Unable to assume the service linked role` (ou une erreur d'attribution de rôle similaire) juste après avoir créé ces rôles, attendez un court moment (environ une minute) que les rôles se propagent, puis réessayez la commande.

Étape 2 : Créez votre première application de service en mode Express

Créez une application de service en mode Express avec les paramètres minimaux requis :

```
aws ecs create-express-gateway-service \
    --primary-container "image"="public.ecr.aws/nginx/nginx:latest" \
    --execution-role-arn arn:aws:iam::123456789012:role/ecsTaskExecutionRole \
    --infrastructure-role-arn arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForExpressServices \
    --monitor-resources
```

Cette commande crée une application de service en mode Express avec :

- Un nom de service unique généré à partir du nom de l'image
- 1 processeur virtuel et 2 Go de mémoire
- Port 80 pour le trafic HTTPS
- Mise à l'échelle automatique en fonction de l'utilisation du processeur
- Un équilibreur de charge des applications connecté à Internet dans le VPC par défaut et les sous-réseaux publics
- URL unique à ce service au format `servicename.ecs.region.on.aws`

La commande surveille en permanence les ressources au fur et à mesure de leur provisionnement et renvoie leur état. Une fois le déploiement du service terminé, l'URL de l'application est prête à recevoir du trafic. Lorsque vous interrompez la surveillance, les informations relatives au service créé sont renvoyées, y compris l'ARN et l'URL du service :

```
{
    "service": {
        "cluster": "arn:aws:ecs:region:123456789012:cluster/default",
        "serviceName": "nginx-1234",
        "serviceArn": "arn:aws:ecs:region:123456789012:service/default/nginx-1234",
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForExpressServices",
        "status": {
            "statusCode": "ACTIVE"
        },
        "createdAt": "UNIXTIMESTAMP"
    }
}
```

Étape 3 : Création d'une application de service en mode Express avec des paramètres personnalisés

Vous pouvez personnaliser votre application de service Express Mode en spécifiant des paramètres supplémentaires :

```
aws ecs create-express-gateway-service \
    --execution-role-arn arn:aws:iam::123456789012:role/ecsTaskExecutionRole \
    --infrastructure-role-arn arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForExpressServices \
    --primary-container '{"image":"123456789012.dkr.ecr.region.amazonaws.com/my-
app:latest","containerPort":8080,"environment":[{"name":"ENV","value":"production"},
{"name":"DEBUG","value":"false"}]}' \
    --service-name "my-web-app" \
    --cpu 2 \
    --memory 4 \
    --cpu-architecture ARM64 \
    --health-check-path "/health" \
    --scaling-target '{"minTaskCount":3,"maxTaskCount":100}' \
    --monitor-resources
```

Cela crée une application avec :

- Un nom personnalisé « my-web-app »
- 2 processeurs virtuels et 4 Go de mémoire
- L'architecture du processeur ARM64, qui fonctionne sur AWS Graviton-powered calculer
- Port 8080 pour le trafic des applications
- Point de terminaison de contrôle de santé personnalisé
- Variables d’environnement
- Minimum 3 tâches, maximum 100 tâches pour la mise à l'échelle automatique

Créez un service en mode Express avec votre propre définition de tâche

Vous pouvez également créer un service en mode Express avec votre propre définition de tâche. Avec votre propre définition de tâche, vous avez le contrôle total de la configuration au niveau des tâches. Le mode Express continue de gérer l'infrastructure, y compris l'équilibrage de charge, la mise à l'échelle automatique et les déploiements Canary.

Les suivants AWS Command Line Interface (AWS AWS CLI) crée un service en mode Express avec une définition de tâche personnalisée :

```
aws ecs create-express-gateway-service \
    --infrastructure-role-arn arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForExpressServices \
    --task-definition-arn arn:aws:ecs:us-west-2:123456789012:task-definition/my-td:3 \
    --health-check-path "/health" \
    --monitor-resources
```

La définition de votre tâche doit répondre aux exigences suivantes :

- Un conteneur nommé `Main`
- Le `Main` le conteneur doit avoir un seul mappage de port TCP avec un port de conteneur et un nom de port
- La définition de la tâche doit avoir `FARGATE` dans ses compatibilités

> **Note**
>
> Le `taskDefinitionArn` le paramètre ne peut pas être spécifié avec `primaryContainer`, `executionRoleArn`, `taskRoleArn`, `cpu`, ou `memory` dans le même appel d'API. Le mode Express dérive ces valeurs à partir de la définition de tâche fournie.

Étape 4 : Surveillez votre déploiement

Le `--monitor-resources` flag fonctionne sur tous les appels de création, de mise à jour ou de suppression vers vos services en mode Express. Mais en plus, vous pouvez surveiller les ressources d'un service à tout moment, indépendamment d'une action de mutation. La durée du déploiement peut varier en fonction des ressources à provisionner. Une fois que le statut passe à `ACTIVE`, votre application est prête à recevoir du trafic.

```
aws ecs monitor-express-gateway-service --service-arn
 arn:aws:ecs:region:123456789012:service/default/app-23d97h88
```

Vous pouvez également consulter la configuration et l'état actuels de votre application de service Express Mode :

```
aws ecs describe-express-gateway-service --service-arn
 arn:aws:ecs:region:123456789012:service/default/app-23d97h88
```

Étape 5 : Accédez à votre application

Accédez à l'application de service Express Mode à l'aide de l'URL fournie lorsqu'elle devient active. Le format de l'URL est le suivant :

```
https://<service-name>.ecs.<region>.on.aws/
```

Par exemple :

```
https://app-23d97h88.ecs.us-west-2.on.aws/
```

Votre application est maintenant en cours d'exécution avec :

- Automatique SSL/TLS résiliation

- Équilibrage de charge entre plusieurs zones de disponibilité
- Mise à l'échelle automatique en fonction de l'utilisation du processeur
- CloudWatch journalisation et surveillance
- Alarmes de restauration 5XX et déploiements de Canary pour les futures mises à jour

Ressources créées par les services Amazon ECS Express Mode

Un service en mode Express réduit la complexité du déploiement d'applications conteneurisées en fournissant des valeurs par défaut judicieuses et en automatisant la configuration du support AWS services. Au lieu de gérer des centaines de paramètres de configuration sur plusieurs services, un service en mode Express ne nécessite qu'une image de conteneur, un rôle d'exécution des tâches et un rôle d'infrastructure pour démarrer.

Les services Express Mode ont simplifié les procédures de provisionnement et de configuration d'un service Amazon ECS qui regroupe l'infrastructure préférée pour faire fonctionner une application Web de production. Les services en mode Express intègrent automatiquement les meilleures pratiques opérationnelles et de configuration. Les ressources suivantes sont créées :

- Le cluster par défaut ECS (s'il n'existe pas déjà) avec les fournisseurs de capacité Fargate
- Une définition de tâche avec des configurations de conteneur, de journalisation et de réseau
- Un service avec déploiement Canary et mise à l'échelle automatique
- Équilibreur de charge des applications avec écouteur HTTPS, règles d'écoute et groupes cibles
- Groupes de sécurité avec entrée minimale requise
- Rôles liés aux services pour la mise à l'échelle automatique et l'équilibrage de charge
- Politique de mise à l'échelle évolutive des cibles et du suivi des cibles par Application Auto Scaling.

  La politique de mise à l'échelle ajoute ou supprime des tâches de service si nécessaire pour maintenir la métrique à la valeur cible spécifiée ou proche de celle-ci. En plus de maintenir la métrique proche de la valeur cible, une politique de suivi des objectifs et d'échelonnement s'ajuste également aux fluctuations de la métrique dues à un modèle de charge fluctuant, et minimise les fluctuations rapides du nombre de tâches exécutées dans votre service.
- Groupe de journaux spécifique au service
- Alarme métrique pour détecter les déploiements défectueux
- Certificat ACM

Utilisez les valeurs par défaut pour commencer à utiliser les services du mode Express ou configurez votre application à l'aide d'un ensemble minimal de paramètres.

Paramètres par défaut du cluster

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour le cluster :

- ClusterName : utilise le `default` grappe.

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en spécifiant `create-express-gateway-service --cluster`.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans le cluster.

- CapacityProviders : ["FARGATE"] - Utilise Fargate

Définition des tâches par défaut

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour la définition des tâches :

- cpu : 1024 - 1 unité vCPU allouée à la tâche

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --cpu`.
- mémoire : 2048 - 2 Go de mémoire allouée à la tâche

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --memory`.
- CPUArchitecture : « X86_64 » : architecture de processeur x86_64 par défaut. Pour utiliser l'architecture ARM 64 bits, choisissez ARM64, qui s'exécute sur AWS Graviton-powered calculer.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans la définition des tâches.

- NetworkMode : « awsvpc » - Utilisations `awsvpc` mode réseau
- opérantSystemFamily: « LINUX » - Fonctionne sur le système d'exploitation Linux

- Compatibilité requise : ["FARGATE"] - Compatible avec le type de lancement Fargate
- PlatformVersion : « LATEST » - Utilise la dernière version de la plateforme Fargate

Vous pouvez également fournir une définition de tâche personnalisée à l'aide du `taskDefinitionArn` paramètre. Le mode Express utilise votre définition de tâche telle quelle, y compris le processeur, la mémoire, les définitions de conteneurs et tous les conteneurs ou paramètres supplémentaires que vous définissez.

Définition des conteneurs par défaut

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour le conteneur :

- port : 80 - Port par défaut pour le trafic de conteneurs

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans la section Définition des tâches - Définition du conteneur :

- essentiel : vrai - Le conteneur est marqué comme essentiel à la santé de la tâche
- protocol : « tcp » - Protocole par défaut pour la communication avec équilibreur de charge
- name : « Main » - Nom par défaut du conteneur principal

  La modification du nom du conteneur par défaut peut affecter la capacité d'Express Mode à effectuer des mises à jour ultérieures de votre service. Cela n'est pas recommandé si vous avez l'intention de continuer à utiliser la console ou les API du mode Express
- VersionConsistency : activé - Résout les balises d'image pour assimiler les ID pour des déploiements cohérents
- initProcessEnabled: true - Active la prise en charge du processus d'initialisation de Docker.
- StopTimeout : 30 secondes - Temps entre les signaux SIGTERM et SIGKILL.

Paramètres de journalisation par défaut

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour CloudWatch Journaux :

- awslogs-group : «/aws/ecs/<cluster>/<name>-#### » - Nom du groupe de journaux dérivé du nom du cluster et du service

Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --primary-container “awsLogsConfiguration”=[{“logGroup”=“/my/logGroup/”}]`.

- awslogs-stream-prefix : « ecs » - Préfixe pour les noms des flux de journaux

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --primary-container “awsLogsConfiguration”=[{“logStreamPrefix”=“myprefix”}]`.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans la section Définition des tâches - Configuration du journal :

- LogDriver : « awslogs » - Utilisations CloudWatch Journaux
- awslogs-region : région actuelle - Utilise la même région que le service Express Mode
- mode : « non bloquant » - Mode de journalisation réglé sur non bloquant
- max-buffer-size : « 25m » - Taille maximale de la mémoire tampon pour les écritures de journal

Lorsque vous fournissez une définition de tâche personnalisée à l'aide du `taskDefinitionArn` paramètre, vous configurez la journalisation dans la définition de la tâche.

Paramètres de service par défaut

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour le service Amazon ECS :

- ServiceName : Service en mode Express → Nom : nom du service (fourni par le client ou dérivé du nom de l'image)

  Le nom ne peut être configuré que lors de la création et ne peut pas être configuré lors des mises à jour.
- cluster : service Express Mode → Cluster - cluster ECS pour ce service (fourni par le client ou par défaut)

  Le cluster ne peut être configuré que lors de la création et n'est pas disponible pour être configuré lors des mises à jour.

- DesiredCount : service en mode express → MinTasks - Nombre souhaité pour le service (la valeur par défaut est 1)

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --scaling-target ‘{“minTaskCount”=3}',`

- Mots clés : Service Express Mode Tags pour AWS étiquetage (fourni par le client)

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --tags`. Les balises ne peuvent être ajoutées que lors de la création d'une nouvelle ressource.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans le service Amazon ECS :

- disponibilitéZoneRebalancing: true - Active le rééquilibrage automatique de l'AZ
- CapacityProviderStrategy: {"base » : 1, « CapacityProvider » : « FARGATE », « weight » : 1} - Utilise Fargate pour la capacité de calcul
- DeploymentConfiguration : Canary par défaut - Le service Express Mode utilise les déploiements Canary

  Notez que la stratégie de déploiement ne peut pas être mise à jour sur les services en mode Express.

- activerECSManagedTags: true - Active ECS-managed balisage des ressources
- santéCheckGracePeriodSeconds: 300 - Période de grâce avant que Scheduler n'examine les bilans de santé ELB ou Lattice (s'aligne sur le délai de grâce par défaut du bilan de santé ELB)
- LaunchType : <Not set>- Utilise des fournisseurs de capacité
- LoadBalancers : configuration de l'équilibreur de charge gérée par ECS

  Notez que les configurations de l'équilibreur de charge ne peuvent pas être mises à jour sur les services en mode Express.

- PlacementStrategy : Non défini - Utilise Fargate au lancement (doit toujours être AZ Spread)
- Version de la plate-forme : DERNIÈRE - Version de la plate-forme Fargate (codée en dur selon la dernière version)
- propagateTags : « SERVICE » - Propage les balises du service vers les tâches

- SchedulingStrategy : « REPLICA » - Les services en mode Express sont des services REPLICA
- TaskDefinition : service Express Mode créé : définition de tâche à utiliser (fournie par le service Express Mode)

Paramètres de configuration réseau par défaut

Les valeurs par défaut suivantes sont configurables depuis le mode Express pour le service Amazon ECS - Configuration réseau :

- réseauConfiguration.Subnets: Si aucun n'est fourni, Express Mode utilisera les sous-réseaux publics par défaut dans le VPC par défaut.

  Le VPC par défaut doit disposer d'au moins deux sous-réseaux publics, dans au moins deux zones de disponibilité, avec au moins 8 adresses IP libres disponibles, par bloc CIDR attribué, par sous-réseau.

  Si vous fournissez des sous-réseaux publics personnalisés, Express Mode fournira un ALB connecté à Internet et activera AssignPublicIP pour vos tâches. Si vous fournissez des sous-réseaux privés (sous-réseaux sans passerelle Internet dans leur table de routage), le mode Express fournira un ALB interne.

  Si vous spécifiez des sous-réseaux, le premier service Express Mode pour un VPC définit les sous-réseaux associés à l'équilibreur de charge connecté à Internet ou à l'équilibreur de charge interne de ce VPC. Les services en mode Express suivants qui sont lancés dans le même VPC doivent disposer de sous-réseaux correspondant aux zones de disponibilité prises en charge par l'équilibreur de charge. C'est pourquoi nous vous recommandons de créer des services en mode Express avec des sous-réseaux provenant de toutes les zones de disponibilité.
- réseauConfiguration.SecurityGroups: Si aucun n'est fourni, le mode Express crée à la fois un groupe de sécurité de service et un groupe de sécurité d'équilibrage de charge

  Le Service Security Group autorise le trafic sortant vers l'Internet public, mais celui-ci peut être limité en fonction d'autres aspects de votre réseau, tels que la configuration de votre sous-réseau.

  Le groupe de sécurité de l'équilibreur de charge Le groupe de sécurité de l'équilibreur de charge autorise le trafic entrant pour une résolution HTTPS et le trafic sortant vers votre groupe de sécurité de service sur le port de conteneur que vous spécifiez (par défaut 80). Et est automatiquement mis à jour pour correspondre au port de conteneur indiqué dans vos appels de création ou de mise à jour en mode Express.

Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --network-configuration '{"securityGroup": ["sg-xxxxxxx"]}'`. Lorsque vous fournissez un groupe de sécurité, vous fournissez un chemin d'entrée supplémentaire vers votre service.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être mises à jour directement dans Amazon ECS Service - Configuration réseau :

- assignerPublicIp: En fonction du type de sous-réseau - Activé pour les sous-réseaux publics afin de garantir que les utilisateurs du VPC par défaut puissent accéder à l'Internet public, nous activons les adresses IP publiques pour chaque tâche par défaut. Cette option est désactivée si vous fournissez un sous-réseau privé, et vous êtes alors responsable de la configuration d'une passerelle NAT si vos tâches nécessitent un accès à Internet.

Valeurs par défaut du rôle IAM

Les rôles IAM suivants sont configurables pour un service en mode Express avec création automatique de rôles liés au service, le cas échéant.

- exécutionRoleArn: Rôle d'exécution de la tâche (paramètre obligatoire)

  Bien que les autorisations requises figurent dans la politique gérée, des politiques et autorisations supplémentaires peuvent être associées. Informations supplémentaires sur le AmazonECSTaskExecutionRolePolicy dans le AWS Guide de référence des politiques gérées
- infrastructureRoleArn: Rôle d'infrastructure pour les services Express Gateway (paramètre obligatoire)

  Bien que les autorisations requises figurent dans la politique gérée, des politiques et autorisations supplémentaires peuvent être associées. Informations supplémentaires sur le AmazonECSInfrastructureRoleforExpressGatewayServices Politique gérée dans le AWS Guide de référence des politiques gérées
- tâcheRoleArn: Rôle de tâche pour les appels depuis le code de l'application conteneur (facultatif)

  Vous pouvez personnaliser cette option à l'aide du Configurations supplémentaires option dans la console, ou en exécutant `create-express-gateway-service --task-role-arn`.

  Vous devez créer le nouveau rôle IAM avant de créer ou de mettre à jour le service Express Mode.

Les rôles IAM suivants sont configurés automatiquement par Express Mode.

- ecsServiceRoleForECS: permet à Amazon ECS de gérer votre cluster
- AWSServiceRoleForElasticLoadBalancing: Appels avec d'autres AWS des services pour vous en votre nom, créés automatiquement par Elastic Load Balancing
- AWSServiceRoleForApplicationAutoScaling_ECSService : Appels autres AWS services en votre nom - créés automatiquement par Application Auto Scaling

Lorsque vous fournissez une définition de tâche personnalisée à l'aide du `taskDefinitionArn` paramètre, le rôle d'exécution de la tâche et le rôle de la tâche sont lus à partir de votre définition de tâche.

Paramètres par défaut de l'application Auto Scaling

Vous pouvez configurer les paramètres suivants de mise à l'échelle automatique des applications en mode Express :

- autoScalingTargetValue: 60 - Pourcentage d'utilisation du processeur cible pour la mise à l'échelle
- autoScalingMetric: « CPUUtilization » - Métrique utilisée pour les décisions de dimensionnement automatique
- désiréMinTaskCount: 1 - Nombre minimum de tâches à gérer
- désiréMaxTaskCount: 20 - Nombre maximum de tâches à adapter (peut être augmenté ou diminué)

Les paramètres suivants peuvent être configurés directement dans Application Auto Scaling :

- Type de politique : »TargetTrackingScaling« - Utilise une politique de mise à l'échelle du suivi des cibles
- désactiverScaleIn: false - Permet de réduire la taille lorsque la demande diminue

Valeurs par défaut de la cible de dimensionnement des applications

Les cibles Application Auto Scaling définissent les limites de ressources et de capacité évolutives pour les opérations de dimensionnement automatique. Les éléments suivants peuvent être modifiés en mode Express :

- MaxCapacity: Service en mode express → MaxTasks - Nombre maximum de tâches que le service pourra atteindre (par défaut à partir de CreateLoadBalancedService ou fourni par le client)
- MinCapacity: Service en mode express → MinTasks - Tâches minimales auxquelles le service sera réduit (par défaut à partir de CreateLoadBalancedService ou fourni par le client)
- ScalableDimension: ECS : service :DesiredCount - La propriété à dimensionner
- Tags : Dérivés à partir des tags transmis CreateLoadBalancedService - Tags de ressource pour cette ressource

Les valeurs par défaut suivantes peuvent être modifiées dans la cible évolutive Application Auto Scaling :

- DynamicScalingInSuspended: false - Indique si la mise à l'échelle est suspendue
- DynamicScalingOutSuspended: false - Indique si la mise à l'échelle est suspendue
- ScheduledScalingSuspended: false - Indique si la mise à l'échelle planifiée est suspendue

Paramètres par défaut de la politique de dimensionnement des applications

Les politiques de dimensionnement mettent en œuvre des algorithmes de suivi des cibles qui surveillent l'utilisation du processeur et ajustent le nombre de tâches pour maintenir des performances optimales. Les éléments suivants peuvent être modifiés à partir du mode Express :

- TargetValue: 60 % - La valeur cible pour la politique de suivi (service Express Mode → scaling-target-value)
- PredefinedMetricType: ECSServiceAverageCPUUtilization - Le type de nom de la métrique suivie (service Express Mode → scaling-metric)

  Le mode Express fournit des mesures de dimensionnement pour l'utilisation moyenne du processeur, l'utilisation moyenne de la mémoire et le nombre de demandes par cible. Le nombre de demandes par cible provient de l'équilibreur de charge des applications et peut être configuré jusqu'à 65 536 demandes par cible et par seconde, ce qui correspond à votre service Amazon ECS dans le groupe cible Application Load Balancer.

Les valeurs par défaut suivantes sont définies par le mode Express mais peuvent être modifiées dans la politique Application Auto Scaling :

- PolicyName: <ServiceName> < »TargetMetric« > - Nom de cette politique de dimensionnement (dérivé du nom du service ECS et de la métrique cible)
- PolicyType: TargetTrackingScaling - Le type de mise à l'échelle en cours (par défaut prédéfini)
- ScalableDimension: « ecs:service :DesiredCount« - La propriété à redimensionner (valeur par défaut prédéfinie)
- DisableScaleIn: false - Indique si la mise à l'échelle est désactivée

Valeurs par défaut de l'équilibreur de charge des applications

Vous pouvez configurer les paramètres par défaut suivants de l'équilibreur de charge des applications en mode Express :

- schéma : dépend des sous-réseaux - Crée un Internet-facing équilibreur de charge avec des adresses IP publiques lorsque des sous-réseaux publics sont fournis. Crée un équilibreur de charge interne avec des adresses IP privées lorsque des sous-réseaux privés sont fournis.
- type d'adresse IP : dépend des sous-réseaux - Crée un équilibreur de charge d'application IPv4 uniquement lorsque les adresses sont IPv4. Quand il y a IPv6-enabled sous-réseaux, un équilibreur de charge d'application à double pile sera créé. Notez que si vous avez IPv6-enabled sous-réseaux, le premier service en mode Express d'un VPC définit l'équilibreur de charge interne ou connecté à Internet pour ce VPC. Pour cette raison, nous vous recommandons de créer votre IPv6-only sous-réseaux d'abord, ou dans un nouveau VPC.

Les valeurs par défaut suivantes sont définies en mode Express et vous pouvez les configurer dans Application Load Balancer :

- mode d'atténuation de la synchronisation : désactivé - L'atténuation de la désynchronisation HTTP est désactivée
- access-logs.enabled : false - La journalisation des accès est désactivée
- listener-configurations.protocol : https - Utilise le protocole HTTPS pour une communication sécurisée
- listener-configurations.port : 443 - Écoute sur le port HTTPS standard
- listener-configurations.rule-type : host-header - Achemine le trafic en fonction des règles d'en-tête de l'hôte

Paramètres par défaut du groupe cible

Vous pouvez configurer les paramètres par défaut du groupe cible Amazon EC2 suivants en mode Express :

- health-check-path : (par défaut «/») Chemin de vérification de l'état du service Express Mode : chemin URL pour les demandes de contrôle de santé
- port : (par défaut 80) - Port sur lequel les cibles reçoivent le trafic
- health-check-port : (par défaut 80) - Port pour les demandes de contrôle de santé

Les valeurs par défaut suivantes sont définies par le mode Express et vous pouvez les configurer dans Amazon EC2 Target Groups :

- protocol : HTTP - Protocole pour acheminer le trafic vers des cibles
- version du protocole : HTTP1 - Version du protocole HTTP pour la communication
- vpc-id : default est l'identifiant VPC - Virtual Private Cloud par défaut pour le groupe cible, mais il sera dérivé des sous-réseaux fournis
- protocole de contrôle de santé : identique à Protocole - Protocole pour les demandes de bilan de santé
- contrôle de santé activé : toujours activé - Les bilans de santé sont automatiquement activés
- health-check-interval -seconds : 30 - Temps entre les bilans de santé des cibles individuelles
- health-check-timeout-seconds : 5 - Durée du délai d'expiration pour les réponses au bilan de santé
- healthy-threshold d-count : 5 - Bilans de santé consécutifs réussis requis pour vérifier l'état de santé
- unhealthy-threshold count : 2 - Des bilans de santé échoués consécutifs sont requis pour détecter un état de santé
- type de cible : ip - Les cibles sont enregistrées par adresse IP
- ip-address-type : ipv4 - Utilise les adresses IPv4 pour l'enregistrement des cibles

Partage des ressources et optimisation des coûts

Les services du mode Express partagent automatiquement les ressources lorsque cela est possible afin d'optimiser les coûts :

- Partage de l'équilibreur de charge - Jusqu'à 25 services en mode Express dans le même VPC peuvent partager un équilibreur de charge d'application. Le mode Express fournira des équilibreurs de charge d'application supplémentaires si nécessaire en fonction de la quantité de services en mode express que vous avez provisionnés. Et à mesure que vous réduisez la quantité de services en mode Express dans votre VPC, le mode Express déprovisionne également la charge d'application non utilisée Balancers.This le partage réduit le coût effectif par application à mesure que vous déployez davantage de services en mode Express.
- Partage de clusters - Les services du mode Express peuvent être regroupés dans des clusters Amazon ECS. Les services en mode Express peuvent également partager des clusters Amazon ECS avec des services Amazon ECS qui ne sont pas gérés par Express Mode.

Afficher les détails d'un service Amazon ECS Express Mode

Vous pouvez consulter les détails de votre service Express Mode pour surveiller son état, sa configuration et ses mesures de performance. La page des détails du service fournit des informations complètes sur l'état actuel de votre service, les tâches en cours, l'historique des déploiements et les mesures opérationnelles.

Pour afficher un service en mode Express (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur le Clusters page, choisissez le nom du cluster qui contient votre service Express Mode.
4. Sur la page des détails du cluster, choisissez Des services onglet.
5. Configurez un filtre pour afficher vos services en mode Express. Pour Type de gestion des ressources par filtre, choisissez ECS.

   Un service en mode Express possède Express badge à côté du nom.
6. Sélectionnez le nom de votre service Express Mode pour accéder à la page détaillée du service correspondant.
7. Consultez les informations de service affichées sur la page des détails du service :

   a. Présentation du service Express - Consultez les informations de base sur les services, notamment l'URL de l'application, l'état du service, les tâches en attente et en cours d'exécution et les déploiements actifs.

b. Observabilité - Afficher CloudWatch mesures relatives à l'utilisation du processeur (min, max, moyenne), à l'utilisation de la mémoire (min, max, moyenne), au temps de réponse cible (p50, p90, p99), RequestCount, statuts 4xx et statuts 5xx.

c. Journaux - Consultez les journaux des conteneurs dans vos tâches.

d. Ressources - Votre AWS Ressources créées pour votre service Express Mode. La vue Chronologie se met à jour automatiquement après une création ou une mise à jour et affiche le statut et les événements. La vue Tableau fournit un tableau consultable avec le statut et les ARN complets de chaque ressource.

e. Déploiements - Suivez l'historique des déploiements et l'état actuel du déploiement, incluez la révision du service et les tâches qui fournissent des informations précieuses lors des mises à jour. Le panneau des événements vous permet de savoir quand votre service en mode Express a atteint un état stable.

f. Configuration - Afficher la configuration des services, y compris le cluster, les rôles, la configuration des conteneurs, le calcul, la mise à l'échelle automatique et la configuration réseau.

g. Balises : gérez vos balises.

Mettre à jour un service Amazon ECS Express Mode

Le service Express Mode simplifie le processus de mise à jour du service en fournissant des options configurables pour la mise en réseau, l'équilibrage de charge et la mise à l'échelle automatique des applications, ainsi qu'en orchestrant les modifications apportées à ces services.

Vous pouvez mettre à jour votre service Express Mode pour modifier les images des conteneurs, ajuster l'allocation des ressources ou modifier les paramètres de configuration. Les mises à jour sont déployées à l'aide de déploiements Canary avec des alarmes de restauration basées sur des alarmes pour maintenir la disponibilité.

Comportement de déploiement

Un service en mode Express utilise les déploiements Canary par défaut pour garantir des mises à jour sécurisées et des annulations rapides :

- Un nouvel environnement est créé et des tâches sont déployées en fonction de vos modifications
- 5 % du trafic est transféré vers le nouvel environnement

- Les annulations basées sur les alarmes se déclenchent si la somme des pourcentages d'erreurs 4xx et 5xx est supérieure à 1 pour 2 points de données en 3 minutes
- Les bilans de santé vérifient que les nouvelles tâches sont saines
- Après 3 minutes de cuisson, 100 % du trafic est transféré vers un nouvel environnement
- Après 3 minutes de cuisson pour la surveillance, les anciennes tâches sont progressivement arrêtées et remplacées

Pour plus d'informations, voir Déploiements d'Amazon ECS Canary

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur le Clusters page, choisissez le nom du cluster qui contient votre service Express Mode.
4. Sur la page des détails du cluster, choisissez Des services onglet.
5. Configurez un filtre pour afficher vos services en mode Express. Pour Type de gestion des ressources par filtre, choisissez ECS.

   Un service en mode Express possède Express badge à côté du nom.
6. Sous Configuration :

   a. Spécifiez l'image à utiliser pour votre application. Pour URI de l'image, entrez l'URI de votre image. Pour parcourir vos images Amazon ECR, choisissez Parcourez les images ECR, puis procédez comme suit :

      i. Pour Dépôt privé, choisissez le référentiel privé Amazon ECR.

      ii. Pour Image, choisissez votre image.

      iii. Choisissez le mode d'identification de l'image. Pour Sélectionnez l'image par, choisissez l'une des options suivantes :

         - AWS vous recommande de choisir Résumé d'images.
         - Pour utiliser la balise, choisissez Balise d'image puis choisissez le tag.

   b. Pour utiliser un registre privé, sélectionnez Registre privé. Ensuite, pour ARN ou nom de Secrets Manager, entrez l'ARN Secrets Manager que vous avez créé dans les prérequis.

c. Pour Rôle d'exécution des tâches, choisissez les rôles ou créez-en un nouveau et actualisez. Vous pouvez mettre à jour le rôle d'exécution des tâches lorsque vous devez ajouter des autorisations supplémentaires.

7. En dessous Configurations supplémentaires, personnalisez votre service.

   a. En dessous Récipient:

      i. Pour Port à conteneurs, mettez à jour le port sur lequel votre application écoute (la valeur par défaut est 80).

      ii. Pour Parcours de contrôle de santé, mettez à jour le chemin pour les bilans de santé (par exemple, `/health`).

   b. En dessous Variables d'environnement, ajoutez des paires clé-valeur pour les variables d'environnement dont votre application a besoin :

      i. Pour Clé, entrez le nom de la variable d'environnement.

      ii. Pour Type de valeur, choisissez Variable d'environnement ou Secret.

      iii. Pour Valeur ou valeur provenant de, entrez la valeur ou la référence.

      iv. Choisissez Ajouter une variable d'environnement pour ajouter d'autres variables si nécessaire.

   c. Pour Commande, entrez éventuellement une commande personnalisée pour remplacer l'instruction Docker CMD.

   d. Pour Rôle de la tâche, ajoutez un rôle IAM qui accorde des autorisations à votre application exécutée dans vos conteneurs. Cela permet à votre application de passer des appels d'API à AWS services.

   e. En dessous Calculer:

      i. Pour CPU, mettez à jour l'allocation de processeur virtuel pour vos tâches (par exemple, 1 processeur virtuel).

      ii. Pour Mémoire, mettez à jour l'allocation de mémoire pour vos tâches (par exemple, 2 Go).

      iii. Pour Architecture du processeur, choisissez X86_64 ou BRAS 64. Choisissez BRAS 64 pour exécuter vos tâches AWS Graviton-powered calculer. Lorsque vous modifiez l'architecture du processeur, un nouveau déploiement démarre et remplace les tâches en cours d'exécution.

   f. En dessous Mise à l'échelle automatique:

i. Pour Indicateur de service ECS, choisissez la métrique sur laquelle vous souhaitez vous baser (par exemple, Utilisation moyenne de la mémoire par le service ECS ou Nombre de demandes par cible).

ii. Pour Valeur cible, entrez la cible de mise à l'échelle (par exemple, 60 ou 1000).

iii. Pour Nombre minimum de tâches et Nombre maximum de tâches, mettez à jour les limites de mise à l'échelle.

g. En dessous Journaux:

i. Pour Amazon CloudWatch groupe de journaux, mettez à jour le nom du groupe de journaux pour les journaux de votre application. Notez que cela ne déplacera pas les journaux existants, mais commencera à écrire des journaux à partir de la nouvelle révision du service.

ii. Pour Amazon CloudWatch préfixe du flux de journal, entrez un nouveau préfixe pour les flux de journaux.

8. Choisissez Mettre à jour pour mettre à jour votre service Express Mode.

Mise à jour avec une définition de tâche personnalisée

Vous pouvez mettre à jour votre service Express Mode avec une définition de tâche personnalisée en spécifiant `taskDefinitionArn` paramètre. Le mode Express utilise votre définition de tâche telle quelle et continue de gérer le reste de l'infrastructure.

Lors des mises à jour suivantes, vous pouvez soit :

- Créez une nouvelle révision de votre définition de tâche et mettez à jour le `taskDefinitionArn`. Le mode Express utilise la nouvelle révision pour le prochain déploiement de service.
- Mettez directement à jour les paramètres du mode Express :
  - Pour les paramètres non liés à la définition de la tâche (tels que `healthCheckPath`, `networkConfiguration`, ou `scalingTarget`), le mode Express applique les modifications sans modifier la définition de la tâche.
  - Pour les paramètres liés à la définition des tâches (tels que l'image, le processeur ou la mémoire), utilisez `primaryContainer` ou `cpu`/`memory`), le mode Express crée une nouvelle définition de tâche gérée avec vos modifications tout en préservant votre autre configuration au niveau des tâches.

Pour effectuer une mise à jour avec une définition de tâche personnalisée (console Amazon ECS)

1. Ouvrez la console Amazon ECS et accédez à votre service Express Mode.
2. Choisissez la flèche déroulante à côté de Service de mise à jour, puis choisissez Mise à jour avec définition de tâche personnalisée.
3. Pour Famille de définitions de tâches, sélectionnez une famille de définitions de tâches existante. Pour créer une nouvelle définition de tâche, choisissez le Définitions des tâches lien.
4. Pour Révision de la définition des tâches, sélectionnez la version à utiliser ou laissez ce champ vide pour utiliser la dernière révision.
5. (Facultatif) Agrandir Configurations supplémentaires pour mettre à jour le chemin de vérification de l'état, la mise en réseau ou les paramètres de dimensionnement automatique.
6. Choisissez Mettre à jour.

Ce qui suit AWS AWS CLI la commande met à jour votre service Express Mode avec une définition de tâche personnalisée :

```
aws ecs update-express-gateway-service \
    --service-arn arn:aws:ecs:us-west-2:123456789012:service/default/my-service \
    --task-definition-arn arn:aws:ecs:us-west-2:123456789012:task-definition/my-td:4
```

> **Note**
>
> Si vous utilisez CloudFormation et ajoutent `taskDefinitionArn` à votre service Express Mode pour la première fois, redéployez d'abord votre stack une fois. Cela garantit la compatibilité avec la restauration.

Supprimer les services Amazon ECS Express Mode

Lorsque vous supprimez un service en mode Express, vous supprimez le service et ses ressources associées, notamment le cluster, l'équilibreur de charge et les autres composants de l'infrastructure.

> **Important**
>
> La suppression d'un service en mode Express interrompt toutes les tâches en cours d'exécution et rend votre application indisponible. Cette action ne peut pas être annulée.

Les ressources suivantes peuvent être supprimées en fonction de votre sélection :

- Le cluster Amazon ECS (si aucun autre service n'est en cours d'exécution)
- Le service Amazon ECS, la définition des tâches et toutes les tâches en cours d'exécution
- Groupe de sécurité des services
- CloudWatch groupe de journaux
- Alarme de métrique
- Certificat ACM
- L'équilibreur de charge des applications (si aucun autre service n'est configuré), le groupe cible, le groupe de sécurité, l'écouteur et la règle d'écoute
- Politique de mise à l'échelle automatique, cible évolutive

Tenez compte des points suivants :

- Assurez-vous d'avoir sauvegardé toutes les données importantes avant de les supprimer
- Envisagez de faire passer le service à zéro tâche avant la suppression afin d'arrêter le trafic en douceur
- Vérifiez les dépendances susceptibles d'être affectées par la suppression

Différence entre la suppression d'un service en mode Express et d'un service

Lorsque vous supprimez un service, Amazon ECS le supprime s'il n'y a aucune tâche en cours d'exécution. Amazon ECS ne supprime pas les groupes cibles associés au service ou au cluster par défaut (s'il a été utilisé). Lorsque vous supprimez un service en mode Express, Amazon ECS supprime le service ainsi que toutes les ressources qui sont distinctes du service en mode Express. Par exemple, si le service de service Express Mode a créé l'équilibreur de charge et est le seul service utilisant cet équilibreur de charge, il supprimera l'équilibreur de charge associé. Si l'équilibreur de charge est partagé, même avec des ressources non gérées par le mode Express, il ne sera pas supprimé.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Sur la page Clusters, sélectionnez le cluster pour le service.
3. Sur la page Clusters, choisissez le cluster.
4. Choisissez l'onglet Services.
5. Sélectionnez les services à supprimer.
6. Sélectionnez Supprimer.
7. Dans le Supprimer la confirmation boîte de dialogue :
   - Entrez **`delete`** pour confirmer la suppression.
8. Choisissez Supprimer pour confirmer.
9. Restez sur la page pour suivre la progression de la suppression et consulter la liste des ressources qui ont été supprimées, ainsi que celles qui ont été conservées et peuvent nécessiter un nettoyage manuel.

Meilleures pratiques pour les services Amazon ECS Express Mode

Découvrez les meilleures pratiques et les recommandations pour utiliser efficacement le service Express Mode dans les environnements de production.

Bonnes pratiques de sécurité

Gestion des secrets

- Utilisez le Gestionnaire de secrets pour les secrets - Stockez les données sensibles dans Secrets Manager (par exemple, un référentiel privé ou des informations d'identification de base de données).

  Pour plus d'informations sur les meilleures pratiques de Secrets Manager, consultez Meilleures pratiques de Secrets Manager dans le Guide de l'utilisateur de Secrets Manager.
- Activer le chiffrement au repos - Assurez-vous que les secrets sont cryptés lorsqu'ils sont stockés dans AWS services.

  L'utilisation d'un service tel que Secrets Manager vous permet de chiffrer à l'aide d'un AWS clé gérée ou fournie par le client.
- Mettre en œuvre une rotation secrète - Utilisez la rotation automatique pour les mots de passe de base de données et les clés d'API.

L'utilisation d'un service tel que Secrets Manager permet de gérer la rotation des secrets pour des services tels qu'Amazon Aurora et Amazon RDS.

Exemple d'utilisation de secrets dans le service Express Mode :

```
aws ecs update-express-gateway-service \
    --primary-container \
        '{"environment"=[{"name"="DB_PASSWORD","value"="arn:aws:secretsmanager:us-
west-2:123456789012:secret:prod/db/password"}, \
        {"name"="API_KEY","value"="arn:aws:ssm:us-west-2:123456789012:parameter/prod/
api-key"}]}' \
```

Sécurité du réseau

- Utiliser des sous-réseaux privés pour les applications sensibles - Déployez des applications qui n'ont pas besoin d'un accès direct à Internet dans des sous-réseaux privés.

  Pour plus d'informations sur les architectures recommandées, consultez Connectez l'application Amazon ECS à Internet.

- Configurer les groupes de sécurité pour qu'ils soient le moins permissifs - Limitez le trafic entrant et sortant aux seuls ports et sources nécessaires.

  Pour restreindre le trafic sortant du groupe de sécurité du service Express Mode, vous pouvez le modifier directement dans la console Amazon EC2 Security Groups en modifiant les règles de trafic sortant, ou utiliser les commandes suivantes :

```
aws ec2 authorize-security-group-egress
    --group-id sg-xxxxxxxx \
    --protocol tcp \
    --port 443 \
    --cidr 0.0.0.0/0

aws ec2 revoke-security-group-egress
    --group-id sg-xxxxxxxx \
    --protocol tcp \
    --port 443 \
```

```
    --cidr 0.0.0.0/0
```

- Activer les journaux de flux Amazon VPC - Surveillez le trafic réseau à des fins d'analyse de sécurité et de dépannage.

  Vous pouvez l'activer dans chaque sous-réseau utilisé par vos applications Express Mode dans la console de sous-réseau VPC, ou utiliser `aws ec2 create-flow-logs --resource-ids subnet-xxx`
- Utiliser AWS WAF pour les applications Web - Protégez-vous contre les attaques et les attaques courantes sur le Web.

  Vous pouvez l'activer en créant une ACL Web, puis en l'associant à l'équilibreur de charge des applications utilisé par votre service Express Mode. Dans la console, créez une ACL Web dans le service WAF & Shield et associez-la à votre équilibreur de charge d'application. Ou utilisez `aws wafv2 create-web-acl` et `aws wafv2 associate-web-acl --resource-arn <alb>`.

Optimisation des performances et du calcul

Dimensionnement des ressources

- Right-size CPU et mémoire - Surveillez les performances des applications et ajustez les allocations de processeur et de mémoire en fonction des habitudes d'utilisation réelles.

  Optimiseur de calcul AWS génère des recommandations pour la taille des tâches et des conteneurs Amazon ECS. Pour plus d’informations, consultez Présentation de Optimiseur de calcul AWS dans le Guide de l’utilisateur Optimiseur de calcul AWS .
- Testez les performances de votre application - Pour vous assurer que votre application fonctionne à grande échelle et avec les seuils de dimensionnement et les allocations de ressources donnés, effectuez des tests de charge.

Configuration de mise à l'échelle automatique

- Définissez des seuils de mise à l'échelle appropriés - Configurez des seuils de processeur ou de mémoire qui déclenchent la mise à l'échelle avant que les performances ne se dégradent.

  Vous pouvez modifier la valeur cible de la métrique de service dans votre console de service Express Mode.

Envisagez d'ajouter une politique de dimensionnement prédictif, en particulier si votre trafic suit un schéma temporel. Pour plus d'informations, consultez Utilisation des tendances historiques pour mettre à l'échelle les services Amazon ECS via une mise à l'échelle prédictive.

- Utiliser plusieurs mesures de mise à l'échelle - Pensez à utiliser à la fois le processeur ou la mémoire et une mise à l'échelle basée sur les demandes pour une mise à l'échelle plus réactive.

  Vous pouvez ajouter plusieurs politiques à un service. Le mode Express en ajoute une par défaut, mais vous pouvez associer des politiques supplémentaires directement à votre service.
- Configurer les limites minimales et maximales des tâches - Fixez des limites raisonnables pour contrôler les coûts et garantir la disponibilité.

  Pour les charges de travail de production, une fois les tests initiaux terminés, nous vous recommandons de les exécuter dans trois zones de disponibilité afin de suivre les meilleures pratiques de disponibilité. Vous pouvez mettre à jour le Nombre minimum de tâches dans la console du mode Express, ou en utilisant `update-express-gateway-service --scaling-target '{“minTaskCount”=3}'`.

Surveillance de l'état

- Mettre en œuvre des bilans de santé significatifs - Créez des points de terminaison de contrôle de l'état qui vérifient les dépendances critiques des applications.

  Vous pouvez mettre à jour le Parcours de contrôle de santé dans la console du mode Express. Ou en utilisant `update-express-gateway-service --health-check-path "/health"`.

  Pour plus d'informations sur l'établissement de bilans de santé pour votre candidature, consultez Mise en œuvre de bilans de
- Gardez les bilans de santé légers - Évitez les opérations coûteuses sur les terminaux de contrôle de santé.

  Les exemples peuvent inclure des appels d'API externes, des opérations gourmandes en processeur ou en mémoire, ou des opérations de longue durée pouvant entraîner un délai d'expiration.
- Utilisez les délais d'attente appropriés - Configurez des délais de contrôle de santé qui permettent des temps de réponse normaux tout en détectant rapidement les défaillances.

Les délais de contrôle de santé pour le mode Express sont configurables sur le groupe cible Application Load Balancer. Dans la console Amazon EC2, accédez au Groupes cibles section et sélectionnez votre groupe cible en mode Express. Sélectionnez le Contrôles de santé Tab et clique Modifier, sous Paramètres de contrôle de santé avancés vous pouvez ajuster le délai d'attente. Ou utilisez `aws elbv2 modify-target-group --target-group-arn <targetgroup> --health-check-timeout`.

- Renvoie les codes d'état HTTP appropriés - Utilisez-en 200 pour une utilisation saine et 4xx/5xx pour les états malsains.

Meilleures pratiques opérationnelles

Surveillance et journalisation

- Bénéficiez d'informations améliorées sur les conteneurs - Utilisation CloudWatch; Enhanced Container Insights pour une surveillance complète de vos applications de service en mode Express.

  Pour plus d'informations, voir Configuration de Container Insights sur Amazon ECS.
- Configurer des métriques personnalisées - Publiez des métriques spécifiques à l'application dans CloudWatch; pour la surveillance de la logique métier.

  Pour plus d'informations, voir Publier des métriques personnalisées dans le CloudWatch Guide de l'utilisateur.
- Configuration de la conservation des journaux - Définissez des périodes de conservation des journaux appropriées pour équilibrer les coûts et les exigences de conformité.

  CloudWatch Les groupes de journaux créés par le mode Express sont configurés pour ne jamais expirer et sont conservés lorsque le service Express Mode est supprimé. Vous pouvez régler ce paramètre dans CloudWatch Groupe de journaux.
- Créez des tableaux de bord et des alertes - Configurer CloudWatch; des tableaux de bord et des alarmes pour une surveillance proactive.

Stratégies de déploiement

- Mettre en œuvre les temps de cuisson - Le mode Express met en œuvre un temps de cuisson variable pour garantir que les déploiements ont le temps de se stabiliser tout en réduisant le rayon

d'action des déploiements problématiques. Si votre application a besoin de plus de temps pour se stabiliser, vous pouvez le configurer dans la définition du service Amazon ECS de votre service Express Mode. Pour plus d'informations, voir Création d'un déploiement Amazon ECS Canary.

- Mettre en œuvre des procédures de restauration - Prévoyez de revenir rapidement aux versions précédentes en cas de problème.

  Des bilans de santé significatifs et des annulations basées sur des alarmes peuvent tous deux contribuer à la restauration. La stratégie de déploiement Canary d'Express Mode combinée à des annulations basées sur des alarmes sur le trafic 4xx et 5xx permet de configurer vos déploiements pour des annulations rapides en cas de code d'application ou de configuration défectueux.

Quand utiliser une définition de tâche personnalisée

Envisagez de fournir votre propre définition de tâche personnalisée uniquement dans les cas suivants :

- Votre équipe dispose de définitions de tâches standardisées avec des agents de sécurité, des sidecars de journalisation ou des configurations de conformité qui doivent être présentes sur tous les services.
- Vous avez besoin de paramètres au niveau des tâches qui ne sont pas directement configurables via le mode Express, tels que les contrôles de l'état des conteneurs, les pilotes de journal personnalisés (par exemple, FireLens), paramètres Linux (`initProcessEnabled`, `tmpfs` supports), ou des conteneurs sidecar supplémentaires.
- Vous utilisez l'infrastructure en tant que code et souhaitez définir la définition de votre tâche dans vos modèles et la transmettre à votre service Express Mode.

Résolution des problèmes liés aux services Amazon ECS Express Mode

Cette section vous aide à identifier et à résoudre les problèmes courants liés au déploiement et à la gestion des services en mode Express.

Problèmes de déploiement

Service bloqué en mode ACTIF ou VIDANGE

Symptômes : DescribeServiceRevisions indique que les ressources sont toujours en cours de provisionnement ou de déprovisionnement. DescribeServices indique que le déploiement n'est pas stabilisé

Causes possibles et solutions :

- Autorisations IAM insuffisantes - Vérifiez que le rôle d'exécution des tâches et le rôle d'infrastructure disposent des autorisations nécessaires, comme indiqué dans leurs politiques gérées respectives.

```
# Check if the role has the required managed policy
aws iam list-attached-role-policies --role-name ecsTaskExecutionRole
```

- Échecs d'extraction d'images - Assurez-vous que l'image du conteneur existe et qu'elle est accessible.

```
# Test image pull manually
docker pull 123456789012.dkr.ecr.us-west-2.amazonaws.com/my-app:latest
```

- Problèmes de connectivité réseau - Vérifiez que les sous-réseaux disposent d'un accès à Internet ou de points de terminaison Amazon VPC pour AWS services.
- Limites de ressources - Vérifiez que votre compte dispose d'une capacité Fargate suffisante et qu'il n'a pas atteint les quotas de service.

Étapes du diagnostic :

1. Utiliser DescribeExpressGatewayService pour obtenir votre révision de service actuelle, suivie de DescribeServiceRevisions pour ServiceRevision pour obtenir l'état du provisionnement ou du déprovisionnement
2. Consultez les événements de service dans la console Amazon ECS pour obtenir des messages d'erreur détaillés.

3. Vérifiez que le port du conteneur a été correctement configuré
4. Vérifiez AWS quotas de service pour Amazon ECS et Fargate.

Échecs de démarrage des tâches

Symptômes : Les tâches ne démarrent pas ou s'arrêtent immédiatement après le démarrage.

Causes courantes :

- Erreurs d'application - L'application conteneur se ferme en raison d'erreurs de configuration ou d'exécution.
- Échecs du bilan de santé - L'application ne répond pas aux contrôles de santé sur le port ou le chemin attendu.
- Contraintes liées aux ressources - Allocation de CPU ou de mémoire insuffisante pour l'application.
- Variables d'environnement ou secrets manquants - La configuration requise n'est pas disponible pour l'application.

Étapes de résolution :

1. Vérifiez les connexions de l'application CloudWatch Journaux, obtenez le nom du groupe de journaux auprès de DescribeServiceRevisions:

```
aws logs describe-log-streams --log-group-name /ecs/express-service-my-app
aws logs get-log-events --log-group-name /ecs/express-service-my-app --log-stream-
name stream-name
```

2. Vérifiez que le chemin de vérification de l'état renvoie le statut HTTP 200.
3. Testez l'image du conteneur localement pour vous assurer qu'elle démarre correctement.
4. Vérifiez et ajustez les allocations de processeur et de mémoire si nécessaire.

Erreurs de définition de tâches personnalisées

La création ou la mise à jour du service échoue avec un `InvalidParameterException` lors de l'utilisation du `taskDefinitionArn` paramètre.

« tâcheDefinitionArn ne peut pas être fourni avec PrimaryContainer, exécutionRoleArn, tâcheRoleArn, processeur ou mémoire. »

Le `taskDefinitionArn` le paramètre ne peut pas être spécifié avec `primaryContainer`, `executionRoleArn`, `taskRoleArn`, `cpu`, ou `memory` dans le même appel d'API. Le mode Express dérive ces valeurs à partir de la définition de tâche fournie.

« La définition de tâche doit contenir un conteneur nommé « Main » avec un seul mappage de port TCP. »

La définition de votre tâche ne répond pas aux exigences du mode Express. Assurez-vous qu'il possède :

- Un conteneur nommé `Main`
- Le `Main` le conteneur a exactement un mappage de port TCP avec un port de conteneur et un nom de port définis
- La définition de la tâche comporte `FARGATE` dans ses compatibilités

Problèmes de connectivité

Application inaccessible via l'équilibreur de charge

Symptômes : L'URL de l'application renvoie des délais d'expiration ou des erreurs de connexion.

Étapes de résolution des problèmes :

1. Validez que le provisionnement de vos ressources est terminé
2. Vérifiez que les tâches sont en cours d'exécution et qu'elles fonctionnent correctement :

```
aws ecs describe-services --cluster my-cluster --services my-express-service
```

3. Vérifiez l'état de santé du groupe cible d'Application Load Balancer :

```
aws elbv2 describe-target-health --target-group-arn
 arn:aws:elasticloadbalancing:region:account:targetgroup/name/id
```

4. Assurez-vous que l'application écoute sur le port approprié à l'intérieur du conteneur.

Problèmes de performance

Temps de réponse lents

Symptômes : Les réponses des applications sont plus lentes que prévu.

Approche diagnostique :

1. Surveillez l'utilisation du processeur et de la mémoire :

```
# Check CloudWatch metrics for the service
aws cloudwatch get-metric-statistics \
    --namespace AWS/ECS \
    --metric-name CPUUtilization \
    --dimensions Name=ServiceName,Value=my-express-service Name=ClusterName,Value=my-
cluster \
    --start-time 2024-01-01T00:00:00Z \
    --end-time 2024-01-01T01:00:00Z \
    --period 300 \
    --statistics Average
```

2. Consultez les journaux des applications pour détecter les erreurs ou les avertissements relatifs aux performances.
3. Vérifiez si la mise à l'échelle automatique répond correctement à la charge.
4. Analysez les métriques de l'équilibreur de charge pour la distribution des demandes.

Stratégies d'optimisation :

- Augmentez l'allocation du processeur ou de la mémoire si les ressources sont limitées.
- Ajustez les seuils de mise à l'échelle automatique pour les réduire plus tôt.
- Optimisez le code de l'application et les requêtes de base de données.

La mise à l'échelle automatique ne fonctionne pas comme prévu

Symptômes : Le service n'augmente pas en cas de charge élevée ni ne diminue en cas de faible charge.

Étapes de résolution des problèmes :

1. Vérifiez les politiques de dimensionnement automatique et leur configuration :

```
aws application-autoscaling describe-scaling-policies \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-express-service
```

2. Révision CloudWatch des mesures pour garantir que les déclencheurs de mise à l'échelle sont atteints.
3. Vérifiez que le service est autorisé à évoluer (vérifiez les rôles IAM).
4. Vérifiez toutes les activités de mise à l'échelle et leurs résultats.

Outils de surveillance et de débogage

En utilisant CloudWatch Informations sur les conteneurs

Activez Container Insights pour une surveillance complète :

```
aws ecs put-account-setting --name containerInsights --value enabled
```

Container Insights fournit :

- Statistiques relatives au processeur, à la mémoire, au disque et au réseau
- Tableaux de bord de suivi des performances
- Corrélation et analyse des logs
- Détection des anomalies

Mise à jour des ressources en dehors du mode Express

Le mode Express simplifie le déploiement en provisionnant et en gérant automatiquement AWS ressources en votre nom, mais il est important de comprendre le modèle de responsabilité. Pendant que le mode Express crée et configure des ressources telles que les services Amazon ECS, les équilibreurs de charge et les politiques de dimensionnement automatique, toutes les ressources restent dans votre AWS compte et sont entièrement accessibles pour une gestion directe. Vous pouvez modifier ces ressources en dehors du mode Express à l'aide du AWS Console, AWS CLI, ou des API, mais cela peut affecter la capacité d'Express Mode à gérer votre service. Cette section explique quelles ressources le mode Express gère, comment apporter des modifications en toute sécurité en dehors du mode Express et ce qui se passe lorsque vous le faites.

Comprendre le modèle de responsabilité partagée

La limite de responsabilité en mode Express est définie par l'utilisation de l'API. Les API Express Mode gèrent les ressources en votre nom grâce à des mises à jour coordonnées entre plusieurs AWS services. Norme AWS Les API de service vous permettent de contrôler directement les ressources individuelles. Cela vous permet d'aller au-delà de l'ensemble de paramètres que le mode Express fournit pour la configuration et vous permet de contrôler chaque paramètre de chaque ressource. Les deux ensembles d'API fonctionnent sur la même infrastructure sous-jacente de votre compte et exécuteront les opérations que vous demandez. AWS a la responsabilité d'effectuer l'opération que vous demandez. Le mode Express ne vérifie pas si les modifications de ressources à l'aide d'API directes entreront en conflit avec la configuration du mode Express, et il n'empêche pas les modifications susceptibles de perturber les fonctionnalités du service. Cela signifie également que vous avez la possibilité d'apporter des modifications, et ces modifications persisteront. Le mode Express ne remplacera pas les modifications, sauf si cela est demandé dans le cadre d'une mise à jour du mode Express. Il vous incombe de comprendre comment les modifications effectuées à l'aide d'API directes interagissent avec la configuration du Mode Express et de résoudre les conflits ou problèmes de service qui en résultent.

Pour plus d'informations sur les valeurs par défaut définies par Express Mode et les ressources qu'il orchestre, consultez the section called “Ressources créées par les services Express Mode ”.

Dois-je faire quelque chose avant de pouvoir profiter des fonctionnalités d'Amazon ECS dans mon service Express Mode ?

Non, il n'y a pas de chemin de graduation ni de verre brisé pour le mode Express. L'ensemble des fonctionnalités d'Amazon ECS est toujours disponible pour votre service Express Mode. Si vous

souhaitez vous assurer qu'aucune autre mise à jour du mode Express ne pourra être effectuée sur votre service, plusieurs options s'offrent à vous. En supprimant le tag géré de vos ressources, Amazon ECS ne sera plus en mesure de les identifier et de les utiliser en tant que service en mode Express. Vous pouvez également utiliser IAM pour restreindre les autorisations des utilisateurs aux API du mode Express.

Exemples de mises à jour en dehors du mode Express

Ajouter un side-car à un service en mode Express

Vous pouvez ajouter n'importe quel conteneur sidecar à votre service. Cet exemple montre comment ajouter un side-car de journalisation Fluent Bit.

Vous pouvez définir une définition de tâche à l'aide de conteneurs sidecar et la fournir à votre service Express Mode à l'aide du `taskDefinitionArn` paramètre. Le mode Express utilise votre définition de tâche telle quelle, y compris tous les conteneurs sidecar que vous avez configurés. Pour plus d'informations, consultez the section called “Créez un service en mode Express avec votre propre définition de tâche”.

Vous pouvez également ajouter un side-car en modifiant directement la définition de tâche créée en mode Express, comme décrit dans la procédure suivante.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sélectionnez le cluster dans lequel se trouve votre service Express Mode, et dans le Services panneau, sélectionnez-le.
4. Accédez au Ressources onglet et sélectionnez la définition de la tâche.
5. Sélectionnez le JSON onglet, puis sélectionnez la flèche vers le bas dans le bouton supérieur intitulé Créer une nouvelle révision. Vous trouverez ici une action appelée Créer une nouvelle révision avec JSON.
6. Ajoutez le conteneur sidecar avant ou après la définition de votre conteneur principal. Pour plus d'informations, voir Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.
7. Si vous avez un side-car de journalisation, comme dans cet exemple, vous souhaiterez modifier `logDriver` dans le contenant principal :

```
"logConfiguration": {
    "logDriver": "awsfirelens"
}
```

8. En supposant les paramètres par défaut d'un service en mode Express, définissez le conteneur principal MemoryReservation sur 1998 et le conteneur Fluent Bit MemoryReservation sur 50. Cela garantit que la réserve de mémoire totale entre les deux conteneurs ne dépasse pas la quantité que vous avez exécutée sur votre tâche Fargate (2048 pour un service en mode Express par défaut).
9. Créez votre nouvelle révision.
10. Cliquez sur le menu déroulant qui dit Déployer et sélectionnez Service de mise à jour.
11. Sélectionnez le Cluster et Service que vous mettez à jour, puis sélectionnez Forcer un nouveau déploiement. La révision de définition de tâche que vous venez de créer doit être sélectionnée automatiquement. Tâche de sélection Update (Mise à jour).
12. Suivez le déroulement de votre déploiement. Les tâches de ce service disposeront désormais de deux conteneurs. Les mises à jour supplémentaires effectuées via la console du mode Express ou les API n'écraseront pas les modifications apportées à la définition de la tâche, sauf si elles sont en conflit direct. Par exemple, dans ce cas, si vous avez attribué à Express Mode un nouveau nom de groupe de journaux et de flux de journaux, le LogDriver sera redéfini en `awslogs`.

Ajouter un domaine personnalisé à votre service

Prérequis : Ce guide part du principe que votre domaine est une zone hébergée gérée par Route 53 avec un certificat ACM. Si votre domaine est hébergé ailleurs, vous aurez besoin d'un CNAME pour pointer vers l'enregistrement DNS de l'Application Load Balancer.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sélectionnez le cluster dans lequel se trouve votre service Express Mode, et dans le Services panneau, sélectionnez-le.
4. Accédez au Ressources onglet et sélectionnez la règle d'écoute.
5. Sélectionnez Action et Modifier la règle.
6. Copiez la valeur d'en-tête Host actuelle (l'URL de votre application), puis Supprimer la règle puisqu'il ne peut y en avoir qu'un de chaque type.

7. Ajouter une condition de type En-tête d'hôte, collez de nouveau l'URL de votre application en tant que valeur de condition d'en-tête Host.
8. Cliquez Ajouter une valeur de condition OR.
9. Entrez votre domaine personnalisé ici.
10. Cliquez Suivant et Enregistrer les modifications.
11. Sélectionnez maintenant Certificats onglet de l'écouteur Application Load Balancer.
12. Cliquez Ajouter un certificat. Ici, vous pouvez importer le certificat vers l'auditeur.
13. Dans votre zone hébergée Route 53, cliquez sur Créer un enregistrement pour créer un enregistrement de routage vers votre service.
14. Sélectionnez un enregistrement de routage simple.
15. Ajoutez un nom d'enregistrement.
16. Sélectionnez un type d'enregistrement.
17. Acheminer le trafic vers un Alias vers l'application et l'équilibreur de charge classique.
18. Sélectionnez votre région.
19. Sélectionnez l'équilibreur de charge utilisé par votre service Express Mode.

Après quelques minutes de propagation, votre service commencera à recevoir du trafic sur le domaine personnalisé. Si vous ajoutez de nombreux certificats à votre équilibreur de charge d'application ou si vous gérez de nombreux services en mode Express à partir de votre équilibreur de charge d'application, vous pouvez envisager d'augmenter la limite de service pour les certificats sur l'équilibreur de charge des applications.

Passerelle Amazon ECS

Amazon ECS Gateway simplifie la mise en réseau d'entrée pour les applications conteneurisées. Il provisionne et gère automatiquement l'infrastructure qui achemine le trafic vers vos services Amazon Elastic Container Service (Amazon ECS). Au lieu de configurer des équilibreurs de charge d'applications, des auditeurs, des groupes cibles, des groupes de sécurité et des certificats pour chaque service, vous définissez les paramètres d'entrée une fois sur une passerelle et vous partagez cette infrastructure entre de nombreux services.

Vous créez une passerelle avec la configuration réseau de votre choix et vous l'associez à un ou plusieurs clusters Amazon ECS. Chaque service opte ensuite en ajoutant un paramètre de connecteur de passerelle. Amazon ECS provisionne et gère ensuite l'infrastructure d'entrée de votre AWS compte, en attribuant à chaque service connecté un nom d'hôte HTTPS unique.

Pour commencer, consultez Création et gestion d'une passerelle. Pour connecter un service à une passerelle qui existe déjà, voir Connectez les services à une passerelle.

Comment fonctionne Amazon ECS Gateway

Amazon ECS Gateway sépare la gestion de l'infrastructure du déploiement des applications. Il prend en charge deux personnages principaux :

- Les équipes de la plateforme possédez la passerelle. Ils créent et configurent des passerelles et les associent aux clusters Amazon ECS. La configuration d'une passerelle s'applique à tous les services qui y sont connectés et inclut des paramètres relatifs au réseau, à la sécurité et aux opérations :
  - Réseautage, par exemple si les services sont accessibles depuis Internet ou uniquement depuis l'intérieur du VPC, et quels sous-réseaux les équilibreurs de charge utilisent
  - Sûreté, par exemple quel groupe de sécurité contrôle le trafic entrant et quel AWS WAF L'ACL Web s'applique
  - Opérations, tels que l'emplacement des journaux d'accès, la capacité de l'équilibreur de charge à autoriser et l'en-tête de requête qui identifie le trafic de test lors des déploiements progressifs.

  Vous ne pouvez pas modifier plusieurs de ces paramètres après avoir créé la passerelle. Pour plus d’informations, consultez the section called “Personnalisation d'une passerelle”.
- Équipes de candidature intégrer leurs services à une passerelle en ajoutant un paramètre de connecteur de passerelle lors de la création ou de la mise à jour d'un service. Le connecteur

nomme le port de conteneur qui reçoit le trafic et peut également définir un filtre de type d'accès et des contrôles de santé.

La configuration de l'entrée via une passerelle se fait en trois étapes :

1. Créez une passerelle. L'équipe de la plateforme crée une passerelle et définit sa configuration d'entrée, y compris les paramètres réseau et de sécurité.
2. Associez la passerelle à des clusters. L'équipe de la plateforme associe la passerelle à un ou plusieurs clusters Amazon ECS, ce qui la met à la disposition des services de ces clusters.
3. Connectez les services. Chaque service s'inscrit individuellement. Une équipe d'application ajoute un connecteur de passerelle lorsqu'elle crée ou met à jour un service, en nommant le port de conteneur qui reçoit le trafic.

Amazon ECS ne fournit aucune infrastructure tant que le premier service n'est pas connecté. Il crée ou réutilise ensuite les ressources partagées, crée les ressources par service et crée une route de passerelle qui dirige le trafic vers le service via une URL HTTPS unique.

Avantages d'Amazon ECS Gateway

Les passerelles simplifient la configuration et la gestion des équilibreurs de charge d'applications pour vos applications. Amazon ECS Gateway offre les avantages suivants :

- Configuration centralisée. Les équipes de la plateforme définissent les paramètres de réseau, de sécurité et de gestion du trafic une seule fois, et chaque service qui se connecte à la passerelle en hérite sur les clusters. Les équipes d'applications accèdent à leurs services sans configurer elles-mêmes de ressources ni avoir besoin d'expertise en matière de réseau.
- Gestion automatisée de l'infrastructure. Amazon ECS fournit et gère les équilibreurs de charge des applications, les auditeurs, les groupes cibles, les groupes de sécurité, les certificats et les noms d'hôte DNS lorsque les services optent pour une passerelle.
- Optimisation des coûts. Partagez les équilibreurs de charge des applications entre plusieurs services en utilisant la même configuration réseau afin de réduire les coûts.
- Infrastructure transparente. Amazon ECS crée toutes les ressources de votre AWS compte, où vous pouvez les consulter et les gérer à l'aide du Console de gestion AWS et le AWS API.

Concepts clés

Le tableau suivant décrit les concepts que vous utilisez lorsque vous utilisez Amazon ECS Gateway.

Concept	Description
Passerelle	Ressource au niveau du compte, dotée de son propre ARN, qui définit la configuration d'entrée et gère l'infrastructure sous-jacente pour les services qui y sont connectés. Les noms de passerelles sont uniques au sein d'un compte et d'une région.
Gateway-cluster association	Lien entre une passerelle et un cluster qui met la passerelle à la dispositi on des services de ce cluster. Les associations sont multiples : une passerelle peut desservir plusieurs clusters et un cluster peut utiliser plusieurs passerelles. Pour plus d'informations, consultez the section called "Limites du service Gateway".
Connecteur Gateway	Paramètre d'activation d'un service qui le connecte à toutes les passerell es compatibles associées à son cluster. Le connecteur spécifie le nom du port de conteneur à exposer et inclut des paramètres facultatifs de type d'accès et de contrôle de santé.
Route de la passerelle	Ressource dotée de son propre ARN qui représente une connexion entre un service et une passerelle. Amazon ECS crée un itinéraire lorsque le service se connecte et le supprime lorsque le service se déconnect e. L'itinéraire indique l'état de la connexion, l'URL du service et les ressources sous-jacentes spécifiques au service.

Infrastructure gérée par Amazon ECS Gateway

Amazon ECS fournit deux catégories d'infrastructures d'entrée sur votre compte.

Type de ressource	Ressources gérées	Description
Infrastructure partagée	Équilibreur de charge des applicati ons, écouteur d'équilibrage de	Ressources qu'une passerelle gère et partage entre plusieurs services Amazon ECS.

Type de ressource	Ressources gérées	Description
	charge, groupe de sécurité de l'équilibreur de charge	
Service-specific infrastructure	AWS Certificate Manager certificat, nom d'hôte DNS, règle d'écoute, groupe cible, groupe de sécurité du service	Ressources créées par Amazon ECS pour chaque route de passerelle afin de diriger le trafic vers le service correspondant.

Amazon ECS utilise le routage basé sur l'hôte pour diriger les demandes provenant d'un équilibreur de charge d'application partagé vers le service approprié. Chaque service reçoit un nom d'hôte généré sous la forme `https://<unique-id>.ecs.<region>.on.aws`. Le domaine varie selon AWS cloison.

Une passerelle ne fournit rien lorsque vous la créez ou que vous l'associez à un cluster. Lorsque le premier service se connecte, Amazon ECS provisionne le premier équilibreur de charge, son écouteur et son groupe de sécurité, puis ajoute les ressources propres à ce service. Au fur et à mesure que de plus en plus de services se connectent, Amazon ECS réutilise un équilibreur de charge doté de capacités, ou en met en service un autre. Jusqu'à 25 services partagent un même équilibreur de charge d'application, et chaque service possède son propre nom d'hôte, son propre certificat et ses propres règles d'écoute.

Amazon ECS étiquette les ressources qu'il gère `AmazonECSManaged=true` et avec la balise système `aws:ecs:gateway-arn`, dont la valeur est l'ARN de la passerelle. Le groupe cible, les règles d'écoute et le groupe de sécurité des services qu'Amazon ECS crée pour chaque route de passerelle comportent également `aws:ecs:gateway-route-arn`. Utilisez ces balises pour rechercher les ressources dans Elastic Load Balancing, Amazon EC2 et AWS Certificate Manager consoles.

Tarification

L'utilisation d'Amazon ECS Gateway est gratuite. Vous ne payez que pour le sous-jacent AWS les ressources mises en service sur votre compte lorsque les services se connectent à une passerelle, y compris l'Application Load Balancer applicable, Amazon CloudWatch les journaux et les statistiques, ainsi que les frais de transfert de données.

Disponibilité

Amazon ECS Gateway est disponible dans toutes les versions commerciales AWS Régions. Il prend en charge les services Amazon ECS qui utilisent le `awsvpc` mode réseau, sur tous les types de lancement : Amazon EC2, Fargate et Amazon ECS Managed Instances. Vous pouvez créer et gérer des passerelles à l'aide de la console Amazon ECS, AWS CLI, le AWS SDK, et CloudFormation.

Limites du service Gateway

Le tableau suivant répertorie les quotas de service pour Amazon ECS Gateway.

Limite	Value
Max Gateway-Cluster Associations par cluster	5
Nombre maximum de clusters par passerelle	10 000
Nombre maximum de connecteurs de passerelle par service	1
Équilibreur de charge maximum de services par application	25
Équilibreurs de charge d'application maximum par passerelle	200

Le sous-jacent AWS les ressources sont également soumises à leurs propres quotas de service. Reportez-vous à Quotas pour les équilibreurs de charge de vos applications.

Considérations

Consultez les points suivants avant d'adopter Amazon ECS Gateway.

Une passerelle sert **`HTTPS`** trafic uniquement. Une passerelle utilise des équilibreurs de charge d'applications avec un routage basé sur l'hôte. Elle dessert donc les applications Web et les API adressées par nom d'hôte. L'auditeur utilise uniquement le `HTTPS` protocole sur le port 443. Aucun autre protocole n'est pris en charge.

Un service doit utiliser le **`awsvpc`** mode réseau. Un service qui utilise le `bridge`, `host`, ou `none` le mode réseau ne peut pas se connecter à une passerelle.

Une passerelle fonctionne au sein d'un seul compte. Une passerelle et ses clusters doivent se trouver dans le même environnement AWS compte. Vous ne pouvez pas associer une passerelle à un cluster d'un autre compte.

Une passerelle ne fonctionne pas avec les services du mode Express. Il propose uniquement les services Amazon ECS.

Un service possède un connecteur de passerelle. Vous ne pouvez pas ajouter de second connecteur à un service. Un connecteur peut toujours connecter le service à plusieurs passerelles.

Un service ne peut pas choisir une passerelle par son nom. Un connecteur connecte le service à chaque passerelle compatible associée à son cluster. Une passerelle est compatible lorsqu'elle correspond au filtre de type d'accès si le service en définit un, dans le même VPC que le service, et couvre chaque zone de disponibilité dans laquelle le service s'exécute. Amazon ECS exclut toutes les passerelles qui ne correspondent pas, et la demande échoue lorsqu'aucune passerelle compatible ne répond aux trois conditions dans le cluster. Pour plus d’informations, consultez the section called “Comment Amazon ECS sélectionne les passerelles compatibles”.

Certains paramètres de passerelle sont immuables après leur création. Vous ne pouvez pas modifier le nom, le type, le schéma, le port d'écoute, le protocole d'écoute, les sous-réseaux ou le type d'adresse IP d'une passerelle après l'avoir créée. Pour modifier l'un de ces éléments, créez une nouvelle passerelle et transférez-y les services. Pour plus d’informations, consultez the section called “Mettre à jour une passerelle”.

Amazon ECS ne consolide pas les équilibreurs de charge. Une passerelle ajoute des équilibreurs de charge lorsque ceux existants sont pleins et les supprime lorsqu'ils sont vides. Il ne déplace pas les services entre les équilibreurs de charge, de sorte que plusieurs équilibreurs de charge partiellement remplis peuvent rester en place une fois les services interrompus.

Amazon ECS ne répare pas les modifications apportées en dehors de celui-ci. Si vous modifiez ou supprimez une ressource gérée par une passerelle, Amazon ECS ne la restaure pas. Le trafic vers les services concernés peut s'arrêter et les déploiements de services ultérieurs, les mises à jour de passerelles et les opérations de nettoyage peuvent échouer. Pour plus d’informations, consultez the section called “Modifications apportées en dehors d'Amazon ECS”.

Création et gestion d'une passerelle Amazon ECS

Une passerelle Amazon ECS est une ressource au niveau du compte qui contient la configuration d'entrée et les normes de sécurité dont héritent les services connectés. L'équipe chargée de la

plateforme crée des passerelles, configure les politiques réseau et les associe aux clusters Amazon ECS cibles. Les équipes d'application optent ensuite pour leurs services et les services se connectent ensuite aux passerelles compatibles du cluster. Pour plus d'informations, consultez the section called "Connectez les services à une passerelle".

La création d'une passerelle ne permet pas de provisionner immédiatement l'infrastructure. Amazon ECS fournit le premier équilibreur de charge d'application uniquement lorsque le premier service se connecte à une passerelle associée.

Conditions préalables

Avant de créer une passerelle, assurez-vous de disposer des éléments suivants :

- Un rôle IAM d'infrastructure qu'Amazon ECS peut assumer pour provisionner et gérer les ressources entrantes.
- Un VPC par défaut dans AWS Région dans laquelle vous créez la passerelle. Si vous ne possédez pas de VPC par défaut, vous pouvez spécifier vos propres sous-réseaux lors de la création de la passerelle.

Rôle IAM d'infrastructure pour Amazon ECS Gateway

Amazon ECS nécessite un rôle d'infrastructure pour créer, configurer et gérer les équilibreurs de charge, les écouteurs, les groupes cibles, les groupes de sécurité et AWS Certificate Manager des certificats sur votre compte.

Créez un rôle IAM avec une politique de confiance qui permet `ecs.amazonaws.com` principal de service pour l'assumer, et joindre le AWS politique gérée `AmazonECSInfrastructureRoleForGatewayHostSharedALB`. Vous pouvez consulter les autorisations accordées par cette politique dans le AWS Référence des politiques gérées.

```
aws iam create-role \
    --role-name ecsInfrastructureRoleForGateways \
    --assume-role-policy-document '{
        "Version": "2012-10-17",
        "Statement": [
            {
                "Effect": "Allow",
                "Principal": {
                    "Service": "ecs.amazonaws.com"
```

```
                },
                "Action": "sts:AssumeRole"
            }
        ]
    }'

aws iam attach-role-policy \
    --role-name ecsInfrastructureRoleForGateways \
    --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRoleForGatewayHostSharedALB
```

L'utilisateur ou le rôle IAM qui crée ou met à jour une passerelle doit être autorisé à transmettre le rôle d'infrastructure à Amazon ECS. Les champs d'application des politiques suivants `iam:PassRole` à ce rôle d'infrastructure spécifique plutôt que de l'attribuer de manière générale.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForGateways"
        }
    ]
}
```

Une équipe chargée de l'application n'a besoin ni de `iam:PassRole` ni les opérations de passerelle visant à doter son service d'un point de terminaison HTTPS. L'équipe n'a pas non plus besoin d'autorisation pour créer des équilibreurs de charge, car ces autorisations résident dans le rôle d'infrastructure.

> **Note**
>
> Les rôles IAM finissent par être cohérents, de sorte qu'un rôle nouvellement créé peut ne pas être utilisable immédiatement. Juste après avoir créé ces rôles, votre premier `create-gateway` l'appel peut échouer avec une erreur telle que `The infrastructureRoleArn could not be assumed. Verify the role exists and its trust policy allows ECS to assume it.` Attendez un court instant (environ une minute) que le rôle se propage, puis réessayez la commande.

> **Important**
>
> Amazon ECS étiquette toutes les ressources qu'il crée `AmazonECSManaged=true`. Si vous modifiez ou supprimez cette balise d'un équilibreur de charge d'application géré, d'un groupe de sécurité ou d'une règle d'écoute, Amazon ECS ne peut plus mettre à jour ni nettoyer cette ressource.

Créer une passerelle

Lorsque vous créez une passerelle, vous devez spécifier un nom unique, l'ARN du rôle d'infrastructure et `hostSharedAlb` objet de configuration. UNE `hostSharedAlb` gateway partage les équilibreurs de charge des applications entre les services et utilise le routage basé sur l'hôte pour diriger le trafic entrant vers les services ECS connectés. Chaque paramètre de la configuration est facultatif, donc `--configuration '{"hostSharedAlb": {}}'` crée une passerelle qui utilise des valeurs par défaut. `CreateGateway` est une opération synchrone : elle renvoie un état de `ACTIVE`, et Amazon ECS ne fournit aucune infrastructure tant que le premier service n'est pas connecté.

Pour créer une passerelle

```
aws ecs create-gateway \
    --name my-gateway \
    --infrastructure-role-arn arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForGateways \
    --configuration '{
        "hostSharedAlb": {
            "scheme": "internet-facing",
            "networkConfiguration": {
                "subnets": ["subnet-abc123", "subnet-def456"]
            }
        }
    }'
```

Sortie :

```
{
    "gatewayArn": "arn:aws:ecs:us-west-2:123456789012:gateway/my-gateway/a1b2c3d4",
    "name": "my-gateway",
    "status": "ACTIVE"
```

```
}
```

Pour confirmer la passerelle et relire sa configuration, exécutez `aws ecs describe-gateway --gateway my-gateway`. Pour plus d'informations, consultez the section called “Afficher l'état de la passerelle”.

Personnalisation d'une passerelle

Amazon ECS sélectionne une valeur par défaut pour chaque paramètre que vous ne fournissez pas. Pour personnaliser une passerelle, définissez les champs facultatifs dans la configuration.

Lorsqu'un paramètre possède un équivalent dans Elastic Load Balancing, les noms et les valeurs des champs correspondent à l'API Elastic Load Balancing.

Un écouteur Gateway utilise `HTTPS` sur le port 443. Aucun autre protocole ou port n'est pris en charge.

Les paramètres suivants sont immuables une fois qu'une passerelle est créée. Pour modifier un paramètre immuable, vous devez créer une nouvelle passerelle, mettre à jour vos services pour vous y connecter et supprimer la passerelle précédente.

- Nom. Le nom doit être unique dans votre compte et peut comporter jusqu'à 255 caractères.
- Sous-réseaux. Spécifiez les sous-réseaux utilisés par les équilibreurs de charge. Par défaut, il s'agit des sous-réseaux de votre VPC par défaut. Les sous-réseaux d'une passerelle déterminent quel VPC et quelles zones de disponibilité elle couvre. Un service se connecte uniquement aux passerelles qui couvrent le VPC et les zones de disponibilité dans lesquelles il s'exécute. C'est donc à vous de décider quels services peuvent utiliser la passerelle. Si votre service ne s'exécute pas dans votre VPC par défaut, définissez des sous-réseaux lors de la création de la passerelle.
- Type d'adresse IP. Choisissez `ipv4`, `dualstack` ou `dualstack-without-public-ipv4`. La valeur par défaut est `dualstack`.
- Schéma. Sélectionnez `internet-facing` ou `internal`. La valeur par défaut est `internet-facing`. Amazon ECS dérive le type d'accès de la passerelle à partir du schéma, donc `internet-facing` donne un type d'accès de `PUBLIC` et `internal` donne `PRIVATE`. Un service se connecte uniquement aux passerelles dont il accepte le type d'accès. Pour plus d'informations, consultez the section called “Connectez les services à une passerelle”.

Vous pouvez modifier les paramètres suivants à tout moment.

- Groupe de sécurité entrant. Fournissez un groupe de sécurité si vous souhaitez contrôler vous-même le trafic entrant vers les équilibreurs de charge. Dans le cas contraire, Amazon ECS en crée un et gère ses règles entrantes.
- Nombre maximal d'équilibreurs de charge. Entrez le nombre d'équilibreurs de charge des applications que la passerelle peut gérer. La valeur par défaut est 10.
- AWS WAF ACL Web. Entrez l'ARN d'une ACL Web à associer à chaque équilibreur de charge géré par la passerelle. Il n'existe aucune valeur par défaut.
- Mode d'atténuation de la désynchronisation HTTP. Choisissez `monitor`, `defensive` ou `strictest`. La valeur par défaut est `defensive`.
- Journaux d'accès. Les journaux d'accès sont désactivés par défaut. Pour les activer, fournissez un compartiment Amazon S3 qui vous appartient.
- Testez les règles de circulation. Entrez l'en-tête qui achemine le trafic de test vers une nouvelle révision de service. Il n'existe aucune valeur par défaut. Une modification s'applique à chaque service lors du prochain déploiement de ce service.
- Protection contre la suppression. La protection contre la suppression est activée par défaut et la passerelle la propage aux équilibreurs de charge qu'elle gère. Vous ne pouvez pas supprimer la passerelle tant que vous ne l'avez pas désactivée.
- Propagation des balises. La passerelle applique ses balises à l'infrastructure qu'elle gère. Pour conserver vos balises uniquement sur la passerelle, réglez la propagation des balises sur `NONE`.

Associer une passerelle à des clusters Amazon ECS

Les services d'un cluster peuvent utiliser une passerelle uniquement après avoir associé la passerelle à leur cluster. Une passerelle et les clusters qu'elle dessert doivent se trouver sur le même site AWS Région et compte. L'association d'une passerelle n'achemine aucun trafic à elle seule. Cela ne change pas la façon dont les services existants reçoivent le trafic. Les services de ce cluster s'inscrivent séparément et commencent à utiliser la passerelle lors de leur prochain déploiement.

Pour associer une passerelle à un cluster

```
aws ecs associate-gateway-with-cluster \
    --gateway my-gateway \
    --cluster my-cluster
```

Les associations sont nombreuses contre plusieurs. Vous pouvez associer une passerelle à un maximum de 10 000 clusters et un cluster à un maximum de 5 passerelles.

Vous gérez les associations uniquement via `AssociateGatewayWithCluster` et `DisassociateGatewayFromCluster`.

Lorsqu'un service doté d'un connecteur de passerelle est déployé, Amazon ECS évalue les passerelles associées à son cluster. Le service se connecte à des passerelles compatibles en fonction de son type d'accès, de son VPC et de ses zones de disponibilité. Planifiez soigneusement la couverture des sous-réseaux, car c'est elle qui détermine la portée d'une passerelle. Une passerelle fournit uniquement les services du VPC et des zones de disponibilité couverts par ses sous-réseaux. Une couverture étroite empêche le reste des services du cluster de l'utiliser. Pour plus d'informations, consultez the section called “Comment Amazon ECS sélectionne les passerelles compatibles”.

> **Note**
>
> Lorsque vous associez une passerelle à un cluster, les services existants ne sont pas affectés avant leur prochain déploiement. L'ajout d'une autre passerelle à un cluster ne connecte pas automatiquement les services déjà en cours d'exécution.

Pour répertorier les associations de clusters d'une passerelle

```
aws ecs list-gateway-cluster-associations --gateway my-gateway
```

La réponse inclut l'ARN de la passerelle, l'ARN du cluster et l'état de l'association. Une association peut avoir les statuts suivants :

- `ACTIVE`: La passerelle est associée au cluster
- `DISASSOCIATING`: L'association est en cours de suppression. Les itinéraires de passerelle existants peuvent continuer à desservir le trafic jusqu'à ce qu'ils soient supprimés.

Il s'agit de statuts d'association et non de statuts de passerelle. Une passerelle possède son propre ensemble de statuts. Vous pouvez filtrer cette liste par passerelle, par cluster ou par statut. Passe `--cluster` pour voir quelles passerelles possèdent un cluster, ou filtrez par statut pour trouver toutes les associations qui existent encore `DISASSOCIATING`. C'est ainsi que vous pouvez vérifier si une dissociation est en attente avant de supprimer une passerelle. Les résultats sont paginés, 100 par page par défaut.

Mettre à jour une passerelle

La mise à jour d'une passerelle modifie les paramètres dont hérite chaque service connecté, y compris les services et l'infrastructure qui le sous-tendent. Certains paramètres ne peuvent pas être modifiés une fois que vous avez créé une passerelle. the section called “Personnalisation d'une passerelle” décrit les paramètres disponibles et identifie ceux que vous ne pouvez pas modifier.

> **Important**
>
> `UpdateGateway` remplace la configuration de la passerelle par celle de votre demande. L'omission d'un paramètre l'efface au lieu de le laisser tel quel. Avant de procéder à la mise à jour, exécutez `describe-gateway` pour récupérer la configuration actuelle et inclure tous les paramètres que vous souhaitez conserver dans la demande, pas seulement celui que vous modifiez.

Étape 1 : Obtenir la configuration actuelle

```
aws ecs describe-gateway --gateway my-gateway
```

```
{
    "gateway": {
        "gatewayArn": "arn:aws:ecs:us-west-2:123456789012:gateway/my-gateway/a1b2c3d4",
        "name": "my-gateway",
        "status": "ACTIVE",
        "accessType": "PUBLIC",
        "deletionProtection": true,
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForGateways",
        "propagateTags": "GATEWAY",
        "configuration": {
            "hostSharedAlb": {
                "scheme": "internet-facing",
                "protocol": "HTTPS",
                "port": 443,
                "maxAlbCount": 10,
                "currentAlbCount": 2,
                "httpDesync": "monitor",
                "networkConfiguration": {
                    "subnets": ["subnet-abc123", "subnet-def456"],
```

```
                    "ipAddressType": "dualstack"
                }
            }
        },
        "tags": [],
        "createdAt": 1760000000.0,
        "updatedAt": 1760400000.0
    }
}
```

Étape 2 : Choisissez les paramètres à inclure

Spécifiez la nouvelle valeur pour le paramètre que vous modifiez et incluez tous les autres paramètres avec leur valeur actuelle. Amazon ECS ne conserve pas un paramètre que vous omettez. Le tableau suivant indique où va chaque paramètre et ce qui se passe si vous ne le modifiez pas.

Paramètre	Où le transmettre	Si vous l'omettez
`infrastructureRoleArn`	Niveau supérieur	La demande échoue (obligatoire)
`configuration.hostSharedAlb`	Niveau supérieur	La demande échoue (obligatoire)
`deletionProtection`	Niveau supérieur	Réinitialiser à `true`
`propagateTags`	Niveau supérieur	Réinitialiser à `GATEWAY`
`networkConfiguration.inboundSecurityGroup`	`hostSharedAlb`	Supprimé. Amazon ECS reprend la gestion des règles entrantes
`webACLArn`	`hostSharedAlb`	Supprimé. L'ACL Web est dissociée
`httpDesync`	`hostSharedAlb`	Réinitialiser à `defensive`

Paramètre	Où le transmettre	Si vous l'omettez
`accessLogs`	`hostSharedAlb`	Supprimé. La journalisation des accès est désactivée
`testTrafficRules`	`hostSharedAlb`	Supprimé
`maxAlbCount`	`hostSharedAlb`	Réinitialiser à `10`

La passerelle `name`, `scheme`, `protocol`, `port`, `subnets`, et `ipAddressType` sont corrigés lorsque vous créez une passerelle et ne peuvent pas être modifiés. `currentAlbCount` est une valeur signalée par Amazon ECS plutôt qu'un paramètre. Ne les incluez pas tous dans la demande.

> **Note**
>
> Le rôle de l'infrastructure est requis dans chaque `UpdateGateway` demande. Une demande qui n'en contient pas renvoie une erreur de validation. Si vous omettez la protection contre la suppression ou la propagation des balises, Amazon ECS les rétablit à leurs valeurs par défaut.

Étape 3 : mettre à jour la passerelle

Cet exemple change `maxAlbCount` à partir de `10` pour `25`. Il comprend également `httpDesync`, qui ne change pas. L'omettre ramènerait la passerelle à la valeur par défaut de `defensive`.

```
aws ecs update-gateway \
    --gateway my-gateway \
    --infrastructure-role-arn arn:aws:iam::123456789012:role/
ecsInfrastructureRoleForGateways \
    --configuration '{
        "hostSharedAlb": {
            "maxAlbCount": 25,
            "httpDesync": "monitor"
        }
    }'
```

Les paramètres immuables décrivent l'infrastructure qu'Amazon ECS ne peut pas reconstruire sans interrompre le trafic. Pour modifier un paramètre immuable, créez une nouvelle passerelle avec la configuration souhaitée, associez-la au cluster, déplacez les services vers celui-ci, puis dissociez et supprimez l'ancienne passerelle.

Lorsque vous modifiez des paramètres mutables avec `UpdateGateway`, Amazon ECS applique les modifications à tous les équilibreurs de charge gérés par la passerelle de manière asynchrone. Les modifications apportées à l'infrastructure partagée prennent effet au fur et à mesure de l'exécution de la mise à jour. La passerelle entre `UPDATE_IN_PROGRESS`. Amazon ECS effectue des mises à jour continues pour tous les ALB actifs. Lorsque les mises à jour sont terminées, le statut devient `UPDATE_COMPLETE`.

Si une mise à jour ne peut pas être appliquée à un ou plusieurs équilibreurs de charge (par exemple, un ARN WAF non valide ou des autorisations IAM manquantes), l'état de la passerelle passe à `UPDATE_FAILED`. Utiliser `describe-gateway` pour inspecter la panne.

Pour vérifier la progression d'une mise à jour

```
aws ecs describe-gateway \
    --gateway my-gateway \
    --query 'gateway.{name:name,status:status}'
```

> **Note**
>
> Pendant la mise à jour d'une passerelle, ses équilibreurs de charge intègrent à la fois l'ancienne et la nouvelle configuration, car Amazon ECS a appliqué la modification à certaines d'entre elles et pas encore à d'autres. Répétez la commande jusqu'à ce que le statut change.

Une mise à jour peut réussir pour certains équilibreurs de charge et échouer pour d'autres. Amazon ECS fait de son mieux pour mettre à jour chaque ressource au lieu de s'arrêter au premier échec. Une passerelle qui rapporte `UPDATE_FAILED` explique le statut dans `statusReason`. Dans `failures`, il enregistre chaque ressource qu'il n'a pas pu mettre à jour, et pourquoi. Chaque entrée nomme la ressource qui a échoué ainsi que les opérations, les codes d'erreur et les messages associés à chaque échec. Une passerelle enregistre les défaillances d'un maximum de 200 ressources, avec jusqu'à 5 erreurs chacune. Résolvez le problème sous-jacent, tel que l'absence d'autorisations IAM ou une ressource modifiée en externe, puis réessayez `update-gateway` opération.

Pour obtenir la description complète d'une passerelle, y compris sa configuration et le nombre d'équilibreurs de charge dont elle dispose actuellement, voir the section called “Afficher l'état de la passerelle”.

> **Important**
>
> Jusqu'à ce qu'une mise à jour rapporte `UPDATE_COMPLETE`, ne partez pas du principe que toutes les ressources gérées possèdent la configuration requise.

Les sections suivantes décrivent les paramètres que vous pouvez modifier. Chacun affiche uniquement sa propre configuration. Ajoutez-le à la configuration que vous avez récupérée à l'étape 1 et incluez le rôle d'infrastructure dans la demande.

Contrôlez le trafic entrant

Par défaut, Amazon ECS crée un groupe de sécurité pour les équilibreurs de charge de la passerelle et autorise le trafic HTTPS entrant provenant de toutes les adresses IP sur le port 443. Pour contrôler vous-même le trafic entrant, définissez votre propre groupe de sécurité. Amazon ECS associe le groupe de sécurité spécifié aux équilibreurs de charge, supprime les règles entrantes existantes qu'il gérait précédemment et arrête de gérer les règles entrantes.

> **Important**
>
> Avant de spécifier un groupe de sécurité entrant, assurez-vous qu'il autorise le trafic requis pour vos services. Dans le cas contraire, les services existants connectés à la passerelle pourraient devenir inaccessibles.

Pour fournir votre propre groupe de sécurité entrant

```
"networkConfiguration": {
    "inboundSecurityGroup": "sg-085c0f8be53138a08"
}
```

Les équilibreurs de charge sont partagés, de sorte que les modifications apportées aux règles des groupes de sécurité entrants peuvent affecter tous les services utilisant ces équilibreurs de charge. Le groupe de sécurité entrant est le seul paramètre de configuration réseau que vous pouvez

modifier après avoir créé une passerelle. Les sous-réseaux et le type d'adresse IP ne peuvent pas être modifiés.

Associez un AWS WAF ACL web

Amazon ECS applique l'ACL Web à chaque équilibreur de charge géré par la passerelle. L'ACL Web protège ensuite chaque service situé derrière la passerelle sans que chaque équipe ne le configure. L'ACL Web doit être une ACL Web régionale dans le même AWS La région comme porte d'entrée.

Pour associer une ACL Web

```
"webACLArn": "arn:aws:wafv2:us-west-2:123456789012:regional/webacl/my-waf/a1b2c3d4"
```

Définissez le mode d'atténuation de la désynchronisation HTTP

Par défaut, les équilibreurs de charge utilisent `defensive` mode, qui protège contre la plupart des contrefaçons de requêtes tout en autorisant les requêtes qui ne sont pas strictement conformes au protocole HTTP. Utiliser `strictest` pour rejeter toute demande non conforme, ou `monitor` pour autoriser toutes les demandes et enregistrer celles qu'il aurait bloquées.

Si vous envisagez `strictest`, commencez par `monitor` afin que vous puissiez voir ce qu'il rejetterait avant de rejeter quoi que ce soit. Pour consulter les résultats, activez les journaux d'accès.

Pour définir le mode d'atténuation de la désynchronisation

```
"httpDesync": "strictest"
```

Activer les journaux d'accès

Les journaux d'accès sont désactivés jusqu'à ce que vous fournissiez un compartiment Amazon S3. Amazon ECS ne crée pas de compartiment pour vous et aucun paramètre distinct n'active la journalisation. La fourniture d'un bucket active les journaux d'accès.

Pour activer les journaux d’accès

```
"accessLogs": {
    "destination": {
        "s3": { "bucket": "my-gateway-logs", "prefix": "alb" }
    }
}
```

Le seau doit être dans le même AWS La région en tant que passerelle et sa politique de compartiment doivent permettre à Elastic Load Balancing d'y écrire. Le rôle d'infrastructure n'est pas impliqué dans ces écritures, donc le fait de lui accorder des autorisations Amazon S3 n'a aucun effet. Les équilibreurs de charge étant partagés, un compartiment reçoit les journaux de chaque service de la passerelle. Utilisez le nom d'hôte dans chaque enregistrement de journal pour identifier le service auquel une demande a été envoyée.

Configuration des règles de trafic de test

Lors d'un déploiement progressif, une nouvelle révision de service doit recevoir du trafic avant que le trafic de production n'y soit transféré. Les règles de trafic de test identifient les demandes qui atteignent la nouvelle révision.

Vous configurez ces règles sur la passerelle plutôt que sur des services individuels, de sorte que chaque service situé derrière la passerelle utilise la même règle. Amazon ECS fait correspondre les demandes par en-tête, et vous devez fournir à la fois le nom de l'en-tête et au moins une valeur.

Pour configurer les règles de trafic de test

```
"testTrafficRules": {
    "header": { "name": "x-test-traffic", "values": ["canary"] }
}
```

Une requête portant `x-test-traffic: canary` atteint la nouvelle révision lors d'un déploiement. Toutes les autres demandes parviennent à la révision de production. La correspondance des en-têtes est le seul mécanisme. Vous ne pouvez pas envoyer un pourcentage du trafic vers la nouvelle révision.

En dehors d'un déploiement, aucune nouvelle révision n'existe. Par conséquent, une demande contenant l'en-tête de test atteint la révision de production comme toute autre demande.

> **Important**
>
> Une modification des règles de trafic de test ne s'applique pas aux services en cours d'exécution lorsque vous l'effectuez. Elle s'applique à chaque service lors du prochain déploiement de ce service.

Amazon ECS ne réécrit pas les conditions de correspondance d'une règle existante. La modification de l'en-tête crée une nouvelle règle pour la nouvelle révision. Un déploiement qui échoue supprime

uniquement la règle qu'il a créée et laisse la règle correspondant à la révision actuelle inchangée. Par conséquent, certains services d'une passerelle peuvent correspondre à un ancien en-tête de test et d'autres correspondre au nouvel en-tête jusqu'à ce que chaque service soit redéployé. Les équipes chargées des applications doivent redéployer leurs services pour sélectionner un nouvel en-tête. Amazon ECS ne redéploie pas ces services. Coordonnez donc un changement d'en-tête avec les propriétaires des services.

Définissez le nombre maximal d'équilibreurs de charge

`maxAlbCount` limite le nombre d'équilibreurs de charge d'applications que la passerelle peut créer. La valeur par défaut est 10.

Pour augmenter le nombre maximal d'équilibreurs de charge

```
"maxAlbCount": 50
```

Deux limites distinctes déterminent le nombre de services qu'une passerelle peut desservir.

- Services par équilibreur de charge. UNE `hostSharedAlb` Gateway place jusqu'à 25 services sur un seul équilibreur de charge d'application. Il s'agit d'une propriété fixe de l'équilibreur de charge d'application avec routage basé sur l'hôte et vous ne pouvez pas la modifier.
- Équilibreurs de charge par passerelle. `maxAlbCount` est le plafond que vous définissez pour le nombre d'équilibreurs de charge que la passerelle peut créer. La valeur par défaut est 10.

Amazon ECS intègre un équilibreur de charge à 25 services, puis en met en service un autre, jusqu'à `maxAlbCount`. Lorsque le dernier service quitte un équilibreur de charge, Amazon ECS le supprime. Ensemble, les deux limites plafonnent une passerelle à 25 x `maxAlbCount` services, ou 250 par défaut.

Augmentez le maximum avant d'en avoir besoin. Si un service s'active alors que chaque équilibreur de charge est plein et que la passerelle est à son maximum, Amazon ECS ne peut pas provisionner l'entrée pour ce service. La panne se produit lors du déploiement de ce service. L'augmentation du maximum par la suite n'entraîne pas une nouvelle tentative de déploiement qui a échoué. Le propriétaire du service doit donc le déployer à nouveau.

Pour voir le nombre d'équilibreurs de charge d'une passerelle par rapport à son maximum, décrivez la passerelle et comparez `currentAlbCount` avec `maxAlbCount`. Pour voir les équilibreurs de charge eux-mêmes et les routes de passerelle sur chacun d'eux, voir the section called “Inspecter l'infrastructure gérée (vue de la plateforme)”.

Dissocier une passerelle d'un cluster

Dissociez une passerelle d'un cluster lorsque vous ne souhaitez plus que les services de ce cluster se connectent à la passerelle. La dissociation empêche de nouvelles connexions de service via cette association. Elle ne supprime pas immédiatement les connexions de service existantes.

> **Important**
>
> Avant de dissocier une passerelle, vérifiez que les services concernés disposent d'un autre chemin d'entrée s'ils doivent rester accessibles. La suppression de routes de passerelle peut rendre les services inaccessibles via cette passerelle.

Pour dissocier une passerelle d'un cluster

```
aws ecs disassociate-gateway-from-cluster \
    --gateway my-gateway \
    --cluster my-cluster
```

L'association entre dans `DISASSOCIATING` état. Pendant que l'association est `DISASSOCIATING` état :

- Les nouveaux services ne peuvent pas se connecter à la passerelle via le cluster.
- Les services existants continuent de recevoir du trafic via leurs passerelles.
- Les services se déconnectent de la passerelle lorsqu'ils sont mis à jour ou redéployés.
- L'association demeure `DISASSOCIATING` jusqu'à ce que toutes les routes de passerelle aient été supprimées.

Amazon ECS ne redéploie pas automatiquement les services lorsqu'une passerelle est dissociée de son cluster. Les équipes chargées des applications doivent mettre à jour ou redéployer leurs services pour se déconnecter de la passerelle dissociée. Si aucune autre passerelle n'est disponible sur le cluster, supprimez le connecteur de passerelle avant de redéployer le service. Lorsque la dernière route de passerelle est supprimée, Amazon ECS supprime l'association.

Afficher les itinéraires de passerelle qui empêchent la dissociation

Utilisez le `list-gateway-routes` commande pour identifier les services qui ont encore des routes de passerelle associées au cluster.

Pour trouver les routes de passerelle bloquant une dissociation

```
aws ecs list-gateway-routes --gateway my-gateway --cluster my-cluster
```

La réponse identifie les routes de passerelle qui sont toujours connectées. S'il reste des itinéraires, mettez à jour ou redéployez les services concernés pour les déconnecter de la passerelle dissociée.

Outre la passerelle et le cluster, vous pouvez filtrer les itinéraires par service, par nom de port de conteneur et par statut. Le filtrage par statut permet de trouver les itinéraires bloqués au lieu d'être encore utilisés. Une route à laquelle mène l'ARN `DescribeGatewayRoute`, qui indique le nom d'hôte et l'infrastructure spécifique au service pour cette connexion. Pour plus d'informations, consultez the section called “Surveiller une passerelle”.

> **Note**
>
> Le filtrage par nom de service uniquement n'est pas autorisé, car les noms de service ne sont uniques qu'au sein d'un cluster. Transmettez un ARN de service complet ou un nom de service avec son cluster.

Annuler la dissociation d'une passerelle

Pendant qu'une association est présente `DISASSOCIATING` état, vous pouvez le réactiver. Associez à nouveau la passerelle au même cluster, et Amazon ECS renvoie l'association à `ACTIVE`.

Pour réactiver une association

```
aws ecs associate-gateway-with-cluster \
    --gateway my-gateway \
    --cluster my-cluster
```

Ce qu'un service conserve dépend de son redéploiement alors que l'association était en train de se dissocier.

- Services qui n'ont pas été redéployés conservent leurs routes de passerelle et leurs noms d'hôte DNS existants, et continuent à servir le trafic sans interruption.
- Services qui ont été redéployés alors que l'association se dissociait ont perdu leurs passerelles. Chacun a besoin d'un autre déploiement pour recevoir à nouveau une route, et cette route porte un nouveau nom d'hôte DNS.

Une fois la dissociation terminée, l'association disparaît définitivement et ne peut pas être réactivée. Si l'association d'origine a déjà été complètement supprimée, pour utiliser à nouveau la même passerelle avec ce cluster, créez une nouvelle association. Les services précédemment connectés reçoivent de nouvelles routes de passerelle, avec de nouveaux noms d'hôte DNS, lors de leur prochain déploiement.

Supprimer une passerelle

Vous pouvez supprimer une passerelle lorsqu'elle n'est plus nécessaire. Avant de supprimer une passerelle, les conditions suivantes doivent être remplies :

- La protection contre la suppression est désactivée.
- La passerelle n'a aucune association de clusters dans le `ACTIVE` ou `DISASSOCIATING` état.

Étape 1 : Désactiver la protection contre la suppression

La protection contre la suppression est activée par défaut. Assurez-vous donc qu'elle est désactivée avant de continuer.

```
aws ecs update-gateway --gateway my-gateway --no-deletion-protection
```

Étape 2 : dissocier la passerelle de chacun de ses clusters

Exécutez la commande suivante pour chaque cluster associé.

```
aws ecs disassociate-gateway-from-cluster \
    --gateway my-gateway \
    --cluster my-cluster
```

Chaque association séjourne `DISASSOCIATING` jusqu'à ce que chaque route de passerelle pour cette passerelle du cluster soit supprimée. Amazon ECS supprime un itinéraire uniquement lorsque le service qui en est propriétaire est redéployé ou supprimé, et il ne redéploie pas de services pour vous. Coordonnez-vous avec les propriétaires de services pour redéployer chaque service.

Pour rechercher les itinéraires sur lesquels une association est ouverte, exécutez la commande suivante.

```
aws ecs list-gateway-routes --gateway my-gateway --cluster my-cluster
```

Étape 3 : Vérifiez qu'aucune association ne subsiste

Amazon ECS supprime une association une fois son dernier itinéraire terminé. Lorsque la liste est vide, vous pouvez supprimer la passerelle.

```
aws ecs list-gateway-cluster-associations --gateway my-gateway
```

Étape 4 : Supprimer la passerelle

```
aws ecs delete-gateway --gateway my-gateway
```

Le `DeleteGateway` le fonctionnement est synchrone. Lorsque l'opération aboutit, Amazon ECS renvoie les informations de passerelle avec un statut de `DELETED`, et les requêtes suivantes pour la passerelle renvoient un `GatewayNotFoundException`. Si les prérequis ne sont pas remplis, l'opération échoue.

Il ne reste plus rien à nettoyer. Amazon ECS supprime la route de passerelle d'un service et ses ressources lorsque celui-ci se déconnecte, et supprime un équilibreur de charge lorsque son dernier service est interrompu.

Connectez les services Amazon ECS à une passerelle

Vous pouvez connecter un service Amazon ECS à une passerelle en configurant un connecteur de passerelle lorsque vous créez ou mettez à jour le service. Un connecteur de passerelle identifie le port de conteneur nommé qui reçoit le trafic. Vous pouvez également configurer des bilans de santé et spécifier les types d'accès à la passerelle que le service peut utiliser.

Amazon ECS connecte le service aux passerelles compatibles associées à son cluster. Pour chaque connexion, Amazon ECS crée une route de passerelle et fournit un nom d'hôte HTTPS unique.

Conditions préalables

Avant d'ajouter un connecteur de passerelle à un service, assurez-vous de disposer des éléments suivants :

- Une passerelle est associée au cluster ECS de votre service et l'association est ACTIVE. Si aucune passerelle n'est associée, Amazon ECS rejette la demande. Vous pouvez donc le savoir immédiatement au lieu d'attendre qu'un service soit déployé sans URL.

- Votre service possède une définition de tâche qui utilise `awsvpc` mode réseau et définit un port nommé sur le conteneur qui doit recevoir le trafic. Pour plus d'informations, voir Options de mise en réseau des tâches Amazon ECS.
- Au moins une passerelle du cluster est compatible avec votre service. Une passerelle est compatible lorsqu'elle se trouve dans le même VPC que les sous-réseaux de votre service, qu'elle correspond à votre filtre de type d'accès et qu'elle couvre toutes les zones de disponibilité dans lesquelles votre service est exécuté.

Connecter un service à une passerelle

Ajoutez un connecteur de passerelle au service et définissez `portName` au port de conteneur qui doit recevoir le trafic dans votre définition de tâche.

Pour connecter un service à une passerelle

```
aws ecs update-service \
    --cluster my-cluster \
    --service my-service \
    --gateway-connectors '[{"portName": "web"}]'
```

Pendant le déploiement, Amazon ECS sélectionne les passerelles compatibles, provisionne ou réutilise l'infrastructure d'entrée requise, crée le certificat, le nom d'hôte, les règles d'écoute et les groupes cibles, et crée une route de passerelle pour chaque connexion. Chaque route possède son propre nom d'hôte HTTPS. Si plusieurs passerelles sont compatibles, le service se connecte à toutes. Un service possède au plus un connecteur de passerelle.

Personnaliser un connecteur de passerelle

Le `portName` est obligatoire. Il fait référence à un port nommé dans votre définition de tâche. Amazon ECS utilise le numéro, le protocole et la version du protocole de ce port pour configurer le groupe cible. Si le nom ne correspond pas à un port dans la définition de tâche utilisée par votre service, la demande échoue. Les paramètres du bilan de santé et un filtre de type d'accès sont facultatifs.

Filtre de type d'accès

Le `filters.accessTypes` restreint les passerelles auxquelles votre service se connecte. Réglez-le sur `PUBLIC`, `PRIVATE`, ou les deux, et omettez-le pour accepter l'un ou l'autre. UNE `PUBLIC` la

passerelle est accessible depuis Internet. UNE `PRIVATE` la passerelle n'est accessible que depuis un réseau qui peut atteindre son équilibreur de charge interne, de sorte qu'aucune demande provenant d'Internet n'arrive. Le filtre contrôle uniquement la sélection de la passerelle. Il ne configure pas les règles de trafic ni les règles de sécurité.

Pour vous connecter uniquement à des passerelles privées

```
aws ecs update-service \
    --cluster my-cluster \
    --service my-service \
    --gateway-connectors '[{
        "portName": "web",
        "filters": { "accessTypes": ["PRIVATE"] }
    }]'
```

Le rétrécissement du filtre a un effet sur le déploiement qu'il commence. Votre service interrompt ses connexions aux passerelles qui ne correspondent plus et leurs URL cessent de fonctionner.

> **Important**
>
> Un service qui ne devrait jamais être accessible depuis Internet doit filtrer sur `PRIVATE`. Sans le filtre, l'association d'une passerelle Internet au cluster expose ultérieurement le service lors de son prochain déploiement.

Surveillance de l'état

Amazon ECS configure des contrôles de santé sur le groupe cible qu'il crée pour chaque route de passerelle et redirige le trafic uniquement vers les cibles qui réussissent. Par défaut, une passerelle envoie une requête HTTP à `/` sur le port qui `portName` noms, toutes les 30 secondes, et attend un `200` réponse dans les 5 secondes. Un service qui répond `200` à `/` ne nécessite aucune configuration de contrôle de santé.

Définissez un chemin différent si `/` est lent ou effectue un véritable travail, comme l'interrogation d'une base de données ou le rendu d'une page complète. Un bilan de santé se répète et fonctionne toutes les 30 secondes pour chaque cible, et une réponse lente peut dépasser le délai d'expiration du bilan de santé et indiquer qu'une cible saine n'est pas saine. Un terminal léger tel que `/health` évite les deux.

Pour définir un parcours de contrôle de santé

```
aws ecs update-service \
    --cluster my-cluster \
    --service my-service \
    --gateway-connectors '[{
        "portName": "web",
        "healthCheck": { "healthCheckPath": "/health" }
    }]'
```

Pour savoir ce que chaque paramètre contrôle, voir Contrôles de santé pour les groupes cibles d'Application Load Balancer dans le Guide de l'utilisateur des équilibreurs de charge d'application.

> **Important**
>
> Une modification du bilan de santé remplace le groupe cible. Vous ne pouvez donc pas en apporter une sur un service qui utilise `ROLLING` stratégie de déploiement. Échec de la demande avec `Target group properties cannot be changed with ROLLING deployment strategy`. Utilisez un `BLUE_GREEN` ou `CANARY` stratégie de déploiement sur le service, ou définissez la configuration du bilan de santé lors de la première connexion au service.

Comment Amazon ECS sélectionne les passerelles compatibles

Lors du déploiement d'un service, Amazon ECS évalue les passerelles associées au cluster du service, jusqu'à 5 par cluster, et restreint ces candidats. Il ignore les passerelles qui ne correspondent pas au lieu d'échouer à votre déploiement.

Amazon ECS répond à trois critères :

Dimension	Comment Amazon ECS détermine l'éligibilité
Type d'accès	La passerelle correspond à celle du connecteur de passerelle `filters.accessTypes` . Si vous omettez le filtre, les deux `PUBLIC` et `PRIVATE` les passerelles sont éligibles.
VPC	La passerelle se trouve dans le même VPC que les sous-réseaux de votre service. Une passerelle dans un autre VPC n'est pas éligible.

Dimension	Comment Amazon ECS détermine l'éligibilité
Zone de disponibilité	La passerelle doit disposer de sous-réseaux configurés dans chaque zone de disponibilité utilisée par les sous-réseaux configurés du service. Une passerelle qui chevauche uniquement certaines zones de disponibilité n'est pas éligible.

Amazon ECS connecte le service à toutes les passerelles éligibles. Une passerelle incompatible n'échoue pas à elle seule au déploiement, si une autre passerelle associée est éligible. S'il ne reste aucune passerelle éligible, la demande échoue car le service ne peut pas établir de connexion de passerelle.

La couverture de la zone de disponibilité doit être complète et non partielle. Une passerelle atteint uniquement les zones de disponibilité où elle possède des sous-réseaux. Une tâche lancée dans une zone de disponibilité que la passerelle ne couvre pas ne passe jamais ses tests de santé et Amazon ECS la remplace. La couverture complète des zones de disponibilité garantit que les tâches lancées dans l'une des zones de disponibilité configurées du service peuvent être enregistrées auprès de cibles saines derrière la passerelle connectée.

Par exemple, considérez deux passerelles, la passerelle 1 (GW1) et la passerelle 2 (GW2), et un service (S1) dans le même VPC :

Ressource	Zones de disponibilité configurées	Éligible au service
GW1	`us-west-2a` , `us-west-2b` , `us-west-2c`	Oui. Il couvre les deux zones de disponibilité du service.
GW2	`us-west-2a` , `us-west-2b`	Non Il ne couvre pas `us-west-2c` .
S1	`us-west-2b` , `us-west-2c`	

En supposant que le filtre de type d'accès correspond également, Amazon ECS connecte le service à Gateway 1 uniquement. Gateway 1 couvre les deux `us-west-2b` et `us-west-2c`, tandis que Gateway 2 ne couvre pas `us-west-2c`. La connexion du service à Gateway 2 peut entraîner le lancement de tâches dans `us-west-2c` pour échouer aux contrôles de santé de l'équilibreur de charge et être remplacé.

> **Important**
>
> Un connecteur de passerelle ne sélectionne pas une passerelle individuelle par son nom ou son ARN. Le service se connecte à chaque passerelle du cluster qui passe les trois tests. Une passerelle couvrant plus de zones de disponibilité que ce dont votre service a besoin constitue la configuration la plus sûre. Pour connecter un service à une passerelle particulière, configurez cette passerelle pour couvrir chaque zone de disponibilité dans laquelle les tâches du service s'exécutent. Coordonnez les modifications apportées aux associations de clusters avec l'équipe de votre plateforme, car elles peuvent affecter plusieurs services.

Comprendre le déploiement des services

La création ou la mise à jour d'un service avec un connecteur de passerelle démarre un déploiement. Amazon ECS crée la route de passerelle et ses ressources spécifiques au service lors de ce déploiement. La première connexion à une passerelle peut prendre plus de temps, car Amazon ECS peut avoir besoin de créer un équilibreur de charge d'application partagé avant de pouvoir créer les ressources de votre service. Les services ultérieurs réutilisent cet équilibreur de charge, son écouteur et son groupe de sécurité.

Lorsque vous mettez à jour un connecteur de passerelle ou que vous déployez le service après la modification d'une association de clusters, Amazon ECS réévalue les passerelles éligibles. Le déploiement établit des itinéraires vers les passerelles nouvellement éligibles et supprime les connexions qui ne correspondent plus.

Comment le trafic de test atteint une nouvelle révision

Lors d'un déploiement progressif, les requêtes contenant l'en-tête et la valeur du trafic de test de la passerelle atteignent la nouvelle révision du service tandis que tout le reste continue d'atteindre la version actuelle. L'équipe de la plateforme définit cet en-tête sur la passerelle, et non sur votre service, de sorte que chaque service situé derrière la passerelle utilise le même. Demandez-leur ce que c'est.

Une modification de l'en-tête ne concerne votre service que lors de son prochain déploiement, de sorte que votre service peut toujours correspondre à un en-tête précédent. Pour plus d’informations, consultez the section called “Configuration des règles de trafic de test”.

Vérifiez la connexion et recherchez l'URL du service

Chaque passerelle éligible produit une connexion distincte et son propre nom d'hôte HTTPS. Ces connexions appartiennent à une révision de service. Un appel renvoie donc toutes les connexions de la révision que vous nommez, ainsi que l'URL, le type d'accès et l'état de chaque connexion. Au cours d'un déploiement, plusieurs révisions existent et chacune comporte ses propres connexions.

Vous obtenez un ARN de révision lors du déploiement du service. `describe-service-deployments` renvoie l'ARN de la révision vers laquelle un déploiement est en train de migrer, sous `targetServiceRevision.arn`. Pour plus d'informations, voir Afficher les détails des révisions du service Amazon ECS.

Pour vérifier une connexion et obtenir l'URL du service

```
aws ecs describe-service-revisions \
    --service-revision-arns arn:aws:ecs:us-west-2:123456789012:service-revision/my-
cluster/my-service/1234567890123456789 \
    --query 'serviceRevisions[0].ecsManagedResources.gatewayConnections'
```

Chaque connexion entraîne `serviceUrl`, le nom d'hôte HTTPS vers lequel envoyer le trafic, aux côtés `accessType`, qui vous indique la distance que ce nom d'hôte atteint, et `status`. Un nom d'hôte prend la forme `https://<unique-id>.ecs.<region>.on.aws`. Le domaine varie selon AWS cloison. L'identifiant du nom d'hôte est aléatoire. Il n'est pas dérivé du nom de votre service. Par conséquent, un service que vous supprimez et recréez sous le même nom reçoit une URL différente.

Lisez `status` à chaque connexion, et attendez qu'une connexion soit signalée `ACTIVE` avant de diriger le trafic client vers son URL. `status` rapports `CREATE_IN_PROGRESS` tandis qu'Amazon ECS fournit et `ACTIVE` une fois que cette connexion est prête, exécutez à nouveau la commande jusqu'à ce que les connexions dont vous avez besoin soient signalées `ACTIVE`. Un service connecté à deux passerelles possède deux connexions à vérifier. Une connexion qui rapporte `CREATE_FAILED` signifie que le provisionnement a échoué au lieu qu'il est toujours en cours d'exécution, et attendre plus longtemps ne changera rien. Pour plus d'informations, consultez the section called “Le nom d'hôte du service ne répond pas”.

Les paramètres de passerelle apparaissent dans `DescribeServiceRevisions`, car les connecteurs changent entre les déploiements et une révision enregistre ce qui a été configuré lors du déploiement. Pour connaître l'état de l'itinéraire, inspecter les ressources et parcourir un itinéraire à travers chaque ressource qui se trouve sur son chemin, voir the section called “Surveillez les itinéraires des passerelles”.

Migrer un service

Pour migrer un service qui possède déjà son propre équilibreur de charge, ajoutez-y un connecteur de passerelle. Amazon ECS fournit une route de passerelle et un nouveau nom d'hôte, sans supprimer ni modifier la configuration existante de votre équilibreur de charge. Les deux chemins d'entrée servent le trafic en même temps. Vous pouvez donc valider l'URL de la passerelle avant de supprimer votre propre équilibreur de charge lors d'une mise à jour ultérieure.

Pour déplacer un service d'une passerelle à une autre, coordonnez vos activités avec l'équipe de votre plateforme. Comme le connecteur ne spécifie pas d'ARN de passerelle, une migration est contrôlée par des associations de clusters et des filtres de type d'accès. C'est également pourquoi un paramètre fixé sur une passerelle, tel que son schéma, ses sous-réseaux, son port d'écoute ou son protocole, ne peut pas être modifié sur place, et créer une passerelle de remplacement et y déplacer des services est le seul moyen d'obtenir une valeur différente.

1. Créez et associez la passerelle de destination. Vérifiez que son type d'accès et son VPC correspondent au service et que ses sous-réseaux couvrent toutes les zones de disponibilité dans lesquelles les tâches du service s'exécutent. Une passerelle qui ne couvre que certaines d'entre elles n'est pas éligible et le service ne s'y connectera pas.
2. Redéployez le service afin qu'il se connecte à la passerelle de destination. Si la passerelle d'origine reste éligible, les deux itinéraires peuvent coexister pendant la transition.
3. Vérifiez que la connexion du service à la passerelle de destination est signalée `ACTIVE`, puis mettez à jour les clients qui adressent l'ancien nom d'hôte.
4. Demandez à l'équipe de la plateforme de dissocier l'ancienne passerelle lorsqu'elle n'est plus nécessaire. Redéployez les services concernés pour supprimer les anciennes routes.

> **Important**
>
> La dissociation d'une passerelle affecte tous les services utilisant cette association de cluster. Planifiez le changement avec les autres équipes chargées de l'application. L'ancien nom d'hôte généré cesse de fonctionner une fois sa route de passerelle supprimée.

Déconnecter un service des passerelles

Pour déconnecter un service de toutes les passerelles

```
aws ecs update-service \
    --cluster my-cluster \
    --service my-service \
    --gateway-connectors '[]'
```

Pour ne conserver que certains types d'accès à la passerelle, mettez à jour `filters.accessTypes` au lieu de cela. Lors du prochain déploiement, Amazon ECS supprime les itinéraires pour les passerelles qui ne correspondent plus. Vous ne pouvez pas utiliser un filtre de type d'accès pour supprimer une passerelle spécifique tout en conservant une autre passerelle éligible du même type d'accès.

Lorsqu'un service se déconnecte, Amazon ECS supprime ses routes de passerelle et ses ressources d'entrée spécifiques au service. La passerelle et ses associations de clusters restent disponibles pour les autres services. La suppression d'un service Amazon ECS connecté entraîne également la suppression de ses connexions de passerelle. S'il ne reste aucun service sur un équilibreur de charge géré, Amazon ECS supprime cet équilibreur de charge. Une association de clusters reste dans le `DISASSOCIATING` état jusqu'à ce que chaque route de passerelle de ce cluster soit supprimée, puis Amazon ECS supprime l'association. Pour plus d'informations, consultez the section called “Dissocier une passerelle d'un cluster”.

La déconnexion prend effet au moment du déploiement qu'il démarre, de sorte que le service continue à gérer le trafic via la passerelle jusqu'à la fin du déploiement.

> **Important**
>
> La suppression du connecteur de passerelle rend le service inaccessible via ses noms d'hôtes de passerelle une fois les routes correspondantes supprimées. Assurez-vous que le service possède un autre chemin d'entrée s'il doit rester accessible.

Surveiller une passerelle

Vous pouvez surveiller Amazon ECS Gateway pour voir l'état d'une passerelle, identifier l'infrastructure qu'elle gère et inspecter les connexions entre les passerelles et les services Amazon ECS. Vous pouvez également utiliser Amazon CloudWatch et Elastic Load Balancing pour surveiller le trafic des requêtes et l'état des cibles recevant ce trafic.

La surveillance des passerelles inclut trois types d'informations :

- Statut de la passerelle et de l'association: si la passerelle est disponible, si une mise à jour de configuration a réussi et quels clusters lui sont associés.
- État de l'itinéraire de la passerelle: si Amazon ECS a provisionné ou mis à jour l'infrastructure d'entrée pour une connexion de service particulière.
- Trafic et santé de la cible: indique si l'équilibreur de charge reçoit des demandes et si les cibles desservant un service sont saines.

Ces statuts sont indépendants. Une passerelle peut être `ACTIVE` même si un itinéraire de passerelle individuel n'a pas été approvisionné. Un itinéraire de passerelle peut être `ACTIVE` même si les cibles de ses applications ne sont pas saines.

Afficher l'état de la passerelle

Utiliser `describe-gateway` pour afficher la configuration et l'état actuel d'une passerelle.

```
aws ecs describe-gateway \
    --gateway my-gateway
```

La réponse inclut l'état de la passerelle, le rôle IAM de l'infrastructure et la configuration. Utilisez-le pour vérifier si une modification de l'infrastructure gérée de la passerelle est terminée. Une fois qu'une mise à jour commence, `statusReason` indique combien d'équilibreurs de charge ont été mis à jour sur le total. Par exemple, 12/12 les équilibreurs de charge sont mis à jour lorsque chaque équilibreur de charge a réussi, ou 10/12 les équilibreurs de charge ont été mis à jour, 2 équilibreurs de charge ont échoué alors que d'autres ne l'ont pas fait. Une passerelle qui n'a jamais été mise à jour n'a pas `statusReason`. `failures` n'apparaît que lorsque la passerelle est en état de défaillance. Inspectez-la pour identifier les ressources concernées. Une entrée d'échec identifie l'ARN de la ressource et l'opération qui a échoué, ainsi qu'un code et un message d'erreur. Pour résoudre un échec de mise à jour, voir the section called “Échecs de mise à jour”. Les associations de clusters et les routes d'une passerelle ne font pas partie de cette réponse.

Pour répertorier les passerelles de votre compte et de votre région, utilisez `list-gateways`:

```
aws ecs list-gateways
```

Pour afficher les clusters Amazon ECS associés à une passerelle, utilisez `list-gateway-cluster-associations`:

```
aws ecs list-gateway-cluster-associations \
    --gateway my-gateway
```

Un `ACTIVE` l'association met à disposition une passerelle vers les services compatibles du cluster. Une association en `DISASSOCIATING` reste jusqu'à ce que les routes de passerelle pour ce cluster soient supprimées.

État de la passerelle, de l'association et de l'itinéraire

Le tableau suivant indique l'opération Amazon ECS à utiliser pour chaque ressource et les valeurs d'état qu'elle indique.

Ressource	opérations d'API	Valeurs de statut	Ce que le statut indique
Passerelle	DescribeGateway ListGateways	ACTIF UPDATE_IN_PROGRESS UPDATE_COMPLETE MISE À JOUR ÉCHOUÉE SUPPRIMÉ	État de configuration de la passerelle et résultat des mises à jour de son infrastructure gérée.
Association de clusters Gateway	ListGatewayClusterAssociations	ACTIF DISSOCIANT	Si l'association peut être utilisée pour de nouvelles connexions de service ou si elle attend la suppression des itinéraires existants.
Route de la passerelle	DescribeGatewayRoute ListGatewayRoutes	ACTIF CREATE_IN_PROGRESS ÉCHEC DE LA CRÉATION	État de provisionnement ou de suppression d'une connexion service-passerelle et de ses ressources d'entrée.

Ressource	opérations d'API	Valeurs de statut	Ce que le statut indique
		UPDATE_IN _PROGRESS MISE À JOUR ÉCHOUÉE DELETE_IN_PROGRESS DELETE_FAILED	

UNE passerelle est `ACTIVE` depuis sa création jusqu'à sa première mise à jour. Une mise à jour le déplace vers `UPDATE_IN_PROGRESS` puis à `UPDATE_COMPLETE` ou `UPDATE_FAILED`. Parmi ceux-ci, `UPDATE_FAILED` est le seul État qui a besoin que vous agissiez : lisez `statusReason` et le `failures` liste pour voir quels équilibreurs de charge n'ont pas été mis à jour et pourquoi. `DELETED` est terminal, et une demande ultérieure pour la passerelle renvoie `GatewayNotFoundException`.

UNE association de clusters de passerelles est `ACTIVE` une fois que vous l'avez créé et `DISASSOCIATING` après l'avoir retiré. Ça reste `DISASSOCIATING` jusqu'à ce que chaque route de passerelle pour ce cluster soit supprimée, donc une longue durée de vie `DISASSOCIATING` est un signe que les services sont toujours connectés, pas que quelque chose a échoué.

UNE route de passerelle suit son propre cycle de vie de création, de mise à jour et de suppression et est `ACTIVE` pendant qu'il transporte du trafic. Les trois `*_FAILED` les états sont ceux où un problème de provisionnement par service apparaît. `list-gateway-routes --status CREATE_FAILED` `UPDATE_FAILED` c'est ainsi que vous les trouvez. Une connexion dans la vue de l'application n'a pas de statut distinct : son `status` est l'état de sa route de passerelle, il utilise donc ces mêmes valeurs.

> **Note**
>
> Une passerelle de reporting `ACTIVE` cela ne signifie pas que tous les services qui le sous-tendent sont sains. L'état de la passerelle décrit la propre configuration de la passerelle. Un déploiement de service qui ne parvient pas à provisionner l'entrée apparaît sur la route de la passerelle et une application défectueuse apparaît dans l'état de santé du groupe cible, ce qui ne modifie pas l'état de la passerelle.

Inspecter l'infrastructure gérée (vue de la plateforme)

Utiliser `describe-gateway-resources` pour identifier les ressources d'entrée qu'une passerelle gère pour vous.

```
aws ecs describe-gateway-resources \
    --gateway my-gateway
```

Pour un `hostSharedAlb` passerelle, la réponse regroupe les ressources par Application Load Balancer. Chaque groupe identifie l'équilibreur de charge, son écouteur, son groupe de sécurité et les routes de passerelle associées. Utilisez cette vue pour déterminer quels équilibreurs de charge appartiennent à une passerelle et quelles connexions de service partagent chaque équilibreur de charge.

La réponse renvoie les ARN des ressources gérées par Amazon ECS, et non l'état actuel des ressources. Si la réponse renvoyait un état, cet état peut être obsolète ou contredire ce que rapporte le service propriétaire. Pour l'état réel d'une ressource, utilisez l'ARN avec la propriété AWS service permettant d'inspecter l'état actuel des ressources. Par exemple, pour décrire un équilibreur de charge d'applications géré, exécutez :

```
aws elbv2 describe-load-balancers \
    --load-balancer-arns arn:aws:elasticloadbalancing:us-
west-2:123456789012:loadbalancer/app/example/0123456789abcdef
```

Vous pouvez inspecter les auditeurs, les règles d'écoute et les groupes cibles de la même manière avec les opérations Elastic Load Balancing pertinentes, et les groupes de sécurité avec Amazon EC2. Comme une passerelle ne rapporte que son propre enregistrement et non une lecture en direct, elle ne détecte pas quand un élément qu'elle gère est modifié en dehors d'Amazon ECS. Pour plus d'informations, consultez the section called “Modifications apportées en dehors d'Amazon ECS”.

Surveillez les itinéraires des passerelles

Utilisez des routes de passerelle pour surveiller les connexions de service individuelles. Chaque route possède son propre statut de provisionnement et une URL de service générée, même lorsque plusieurs itinéraires partagent un équilibreur de charge d'application.

Liste des itinéraires de passerelle

Pour voir toutes les connexions de service via une passerelle, exécutez :

```
aws ecs list-gateway-routes \
    --gateway my-gateway
```

Pour trouver les itinéraires d'un service particulier, spécifiez le cluster et le service :

```
aws ecs list-gateway-routes \
    --cluster my-cluster \
    --service my-service
```

La réponse identifie l'ARN de la route, le service et la passerelle associés, ainsi que l'état de la route. Vous pouvez filtrer les itinéraires par passerelle, cluster, service, nom de port de conteneur ou statut. Pour filtrer par service, transmettez un ARN de service complet ou un nom de service avec son cluster. Un nom de service en lui-même est ambigu, car les noms de service ne sont uniques qu'au sein d'un cluster. Amazon ECS renvoie donc une erreur. Utilisez le filtre d'état pour rechercher les itinéraires dont le provisionnement ou la mise à jour a échoué :

```
aws ecs list-gateway-routes \
    --gateway my-gateway \
    --status CREATE_FAILED UPDATE_FAILED
```

Un itinéraire dans `CREATE_IN_PROGRESS` fournit toujours des ressources d'entrée et une route `ACTIVE` est terminé. Si un itinéraire est en état de défaillance, inspectez-le avec `describe-gateway-route`, et voyez the section called “Résoudre les problèmes liés à une passerelle” pour obtenir des conseils en matière de rétablissement.

Inspecter un itinéraire de passerelle

Pour inspecter une connexion particulière, exécutez `describe-gateway-route` avec la route ARN :

```
aws ecs describe-gateway-route \
    --gateway-route-arn arn:aws:ecs:us-west-2:123456789012:gateway-route/my-gateway/
a1b2c3d4/9f8e7d6c
```

La réponse inclut l'état de la route et l'URL du service, ainsi que les identifiants des ressources d'entrée pour cette connexion : le certificat, l'équilibreur de charge et son écouteur, la règle d'écoute pour le trafic de production, la règle d'écoute pour le trafic de test, les groupes de sécurité de l'équilibreur de charge et le groupe de sécurité de service où la passerelle gère la règle d'entrée.

Utilisez ces identifiants pour suivre une demande de l'équilibreur de charge partagé adressée au service. Une route répertorie normalement une règle d'écouteur de test, et en répertorie d'autres uniquement lorsqu'elle passe d'une configuration d'en-tête de test à une autre. Pour trouver le groupe cible qui reçoit le trafic de production ou le groupe cible alternatif utilisé lors d'un déploiement progressif, inspectez plutôt la révision du service.

Afficher les connexions de passerelle pour une révision de service (vue de l'application)

Les équipes chargées de l'application peuvent inspecter les connexions de passerelle enregistrées pour une révision du service Amazon ECS déployé. Utiliser `describe-service-revisions`:

```
aws ecs describe-service-revisions \
    --service-revision-arns arn:aws:ecs:us-west-2:123456789012:service-revision/my-
cluster/my-service/1234567890123456789 \
    --query 'serviceRevisions[0].ecsManagedResources.gatewayConnections'
```

Chaque connexion identifie la passerelle et la route, le nom du port du conteneur, le type d'accès, l'état de la connexion et l'URL du service HTTPS générée. Une connexion `status` est l'état de sa route de passerelle, il indique donc les valeurs d'état de la route de passerelle. Une connexion nomme également les ressources d'entrée pour ce service : le certificat, l'équilibreur de charge et son écouteur, les règles d'écoute pour le trafic de production et de test, les groupes de sécurité de l'équilibreur de charge et les groupes cibles de production et alternatifs. Un déploiement progressif enregistre la nouvelle révision dans le groupe cible alternatif tandis que la révision actuelle continue de fonctionner à partir de la version de production. Inspectez une révision de service lorsque vous souhaitez vérifier les connexions avec lesquelles un déploiement particulier a été configuré. La comparaison de deux révisions montre ce qui a changé en ce qui concerne l'entrée d'un service entre les déploiements. Pour obtenir des instructions sur la création, la modification ou la suppression de connexions de service, voir the section called “Connectez les services à une passerelle”.

Surveillez le trafic et l'état de santé des cibles

Après avoir connecté un service, vous observez deux choses : le trafic qui l'atteint et l'état de santé des cibles qui le desservent. Amazon ECS Gateway ne publie pas ses propres statistiques à cet effet. Vous utilisez plutôt les métriques Application Load Balancer qu'Elastic Load Balancing publie sur Amazon CloudWatch, et vous pouvez lire l'état de santé de la cible directement depuis Elastic Load Balancing.

Choisissez les bonnes dimensions métriques

Une passerelle peut placer jusqu'à 25 services sur un seul équilibreur de charge d'application, de sorte qu'une métrique rapportée pour l'ensemble de l'équilibreur de charge combine tous les services qu'il contient. Pour voir un seul service, interrogez la métrique avec la dimension du groupe cible au lieu de la dimension de l'équilibreur de charge. Choisissez les dimensions qui correspondent à ce que vous voulez voir.

Champ de surveillance	Dimensions recommandées	Métriques utiles
Connexion de service individuelle	`TargetGroup` avec `LoadBalancer`	`RequestCount` , `TargetResponseTime` , `HTTPCode_Target_5XX_Count` , `HealthyHostCount` , `UnHealthyHostCount`
Équilibreur de charge partagé	`LoadBalancer`	`HTTPCode_ELB_5XX_Count` , `ActiveConnectionCount` , `RejectedConnectionCount` , `ConsumedLCUs`
Zone de disponibilité	`AvailabilityZone` avec les dimensions de l'équilibreur de charge ou du groupe cible applicables	Statistiques de santé et de requêtes cibles, afin de déterminer si un problème est limité à une seule zone de disponibilité

Les équipes chargées de l'application doivent utiliser la dimension du groupe cible. Un pic de demandes ou une augmentation du nombre de réponses 5XX sur la dimension de l'équilibreur de charge peuvent provenir d'un autre service qui partage l'équilibreur de charge, et vous ne pouvez pas l'attribuer au vôtre. Trouvez le groupe cible de votre service dans sa connexion de passerelle, qui `describe-service-revisions` retours. Pour la liste complète des mesures et des statistiques prises en charge par chacune d'entre elles, voir CloudWatch des mesures pour votre équilibreur de charge d'application dans le Guide de l'utilisateur des équilibreurs de charge d'application.

> **Note**
>
> Elastic Load Balancing publie une métrique uniquement lorsque le trafic circule et ne prend pas en compte les demandes de contrôle de santé. Un service récemment connecté qui n'a pas encore reçu de trafic n'affiche aucune métrique de demande. C'est normal et cela n'indique pas un problème.

Vérifiez la santé de la cible

Les indicateurs montrent les tendances au fil du temps. Pour évaluer l'état de santé de vos cibles dès maintenant, lisez directement le groupe cible :

```
aws elbv2 describe-target-health \
    --target-group-arn arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/
example/0123456789abcdef
```

La réponse indique l'état actuel de chaque cible enregistrée et la raison pour laquelle une cible n'est pas saine, comme l'échec du contrôle de santé et le code de réponse qu'elle a reçu. Une passerelle vers `ACTIVE` L'état signifie qu'Amazon ECS a terminé le provisionnement de l'entrée. Cela ne signifie pas que les cibles sous-jacentes sont saines, alors vérifiez l'état de santé de la cible séparément. Si les cibles restent malsaines, voir the section called “Connectivité des services et santé de la cible”.

Configurez CloudWatch alarmes

Créez des alarmes sur les indicateurs importants pour votre application, tels que les cibles défectueuses, l'augmentation du temps de réponse cible et les erreurs 5XX cibles. Les équipes de la plateforme peuvent également signaler les erreurs 5XX de l'équilibreur de charge et les connexions rejetées pour l'infrastructure partagée.

Une passerelle ajoute des équilibreurs de charge lorsque les services se connectent et les supprime lorsque les services se déconnectent. Ainsi, une alarme liée à un seul équilibreur de charge ou à l'ARN d'un groupe cible peut arrêter discrètement de mesurer quoi que ce soit lorsque cette ressource disparaît. Amazon ECS étiquette chaque équilibreur de charge qu'il crée avec l'ARN de la passerelle. Sélectionnez les ressources en fonction de cette étiquette afin que vos alarmes et vos tableaux de bord continuent de suivre la passerelle au fur et à mesure de l'évolution de son infrastructure. Pour plus d'informations, voir Utilisation de balises de ressources pour la télémétrie dans le Amazon CloudWatch Guide de l'utilisateur. Tenez compte des métriques qui ne sont pas émises à chaque période en choisissant la manière dont une alarme traite les données manquantes.

Afficher les journaux de trafic et d'exploitation

Journaux d’accès d’Application Load Balancer

Lorsque la journalisation des accès est activée pour les équilibreurs de charge gérés d'une passerelle, Elastic Load Balancing transmet les enregistrements des demandes au compartiment Amazon S3 configuré. Les journaux d'accès capturent des informations sur les demandes et les réponses qui vous permettent d'examiner les modèles de trafic et d'étudier les demandes individuelles.

Les équilibreurs de charge étant partagés, les journaux d'accès d'un même équilibreur de charge peuvent inclure des demandes pour plusieurs services. Utilisez le nom d'hôte dans chaque enregistrement de journal pour différencier les demandes d'un service de celles d'un autre. Les journaux d'accès sont désactivés par défaut. Pour savoir comment les activer, voir the section called “Activer les journaux d'accès”. Pour le format du journal et le comportement de livraison, voir Journaux d'accès pour votre équilibreur de charge d'application dans le Guide de l'utilisateur des équilibreurs de charge d'application.

Localisez les détails des défaillances

Lorsqu'une opération de surveillance signale une panne, utilisez le tableau suivant pour trouver où la panne est enregistrée.

Défaillance constatée	Où chercher	Informations disponibles
Rapports de mise à jour de la configuration d'une passerelle `UPDATE_FAILED`	`describe-gateway`	`statusReason` et `failures` identifier les ressources gérées concernées et les opérations qui n'ont pas abouti.
Un rapport de connexion à un service `CREATE_FAILED` ou `UPDATE_FAILED`	`describe-gateway-route`	L'état de la route et les ressources d'entrée pour cette connexion.
Le déploiement d'un service n'atteint pas l'état attendu	`describe-service-revisions`	Les connexions de passerelle de la révision du service et le contexte des opérations de déploiement.

Défaillance constatée	Où chercher	Informations disponibles
Les demandes échouent ou les cibles de l'application ne sont pas saines	État de santé de la cible Elastic Load Balancing, CloudWatch métriques et journaux d'accès (si activés)	Les états de santé des cibles, les indicateurs de demande et les informations relatives aux demandes individuelles.

Utilisez ces sources pour déterminer quelle ressource ou quelle connexion est affectée. Pour connaître les symptômes, les causes probables et les procédures de guérison, voir the section called “Résoudre les problèmes liés à une passerelle”.

Résoudre les problèmes liés à une passerelle

Utilisez cette rubrique pour résoudre les problèmes liés à Amazon ECS Gateway, notamment les échecs de connexion au service, les échecs de provisionnement d'itinéraires de passerelle, les mises à jour de configuration de passerelle et les problèmes de connectivité.

Les passerelles, routes de passerelle et cibles d'applications Amazon ECS signalent leur état de manière indépendante. Une passerelle peut rester `ACTIVE` même lorsqu'une route de passerelle n'est pas provisionnée et qu'une route de passerelle peut être `ACTIVE` même lorsque les cibles de ses applications ne sont pas saines. Pour identifier la cause d'une panne, examinez l'erreur renvoyée par l'opération de l'API ou examinez l'état et les détails de la panne de la passerelle ou de la route de passerelle concernée.

Avant de poursuivre vos recherches, vérifiez que vous utilisez le nom de ressource ou l'ARN correct, AWS compte et région. Assurez-vous qu'Amazon ECS Gateway est pris en charge dans votre région et que vous disposez des autorisations requises. Pour les problèmes de connexion au service, vérifiez que la définition de tâche du service utilise le `awsvpc` mode réseau.

Si le problème persiste, utilisez les procédures de dépannage suivantes pour identifier et résoudre la panne.

Défaillances de connexion et de déploiement des services

Aucune passerelle n'est associée au cluster

UNE `CreateService` ou `UpdateService` Une requête qui inclut un connecteur de passerelle échoue lorsqu'aucune passerelle n'est associée au cluster du service. Rapports Amazon ECS `gatewayConnectors were specified but no gateways are associated with cluster` et rejette la demande au lieu de déployer un service auquel il ne peut pas donner d'URL. Vérifiez les associations de passerelles du cluster pour vérifier qu'au moins une passerelle est associée au cluster du service.

```
aws ecs list-gateway-cluster-associations \
    --cluster my-cluster
```

Si aucune association n'est disponible, contactez l'équipe de votre plateforme pour associer une passerelle avant de déployer le service avec un connecteur. Une passerelle dont l'association est entrée `DISASSOCIATING` n'est pas disponible pour les nouvelles connexions. Cette erreur s'affiche également lorsque l'équipe de la plateforme a supprimé la dernière passerelle de votre cluster. Supprimez le connecteur de votre service ou attendez une passerelle de remplacement avant votre prochaine mise à jour.

Aucune passerelle compatible ne correspond au service

Amazon ECS évalue les passerelles associées au cluster et connecte le service à chaque passerelle qui répond à toutes les conditions suivantes :

- Type d'accès. Le type d'accès de la passerelle correspond à `filters.accessTypes` réglage dans le connecteur de passerelle du service. Si vous omettez ce filtre, les deux `PUBLIC` et `PRIVATE` les passerelles sont éligibles.
- PVC. La passerelle et le service se trouvent dans le même VPC.
- Zones de disponibilité. La passerelle couvre toutes les zones de disponibilité utilisées par les sous-réseaux configurés du service. Les ID de sous-réseau eux-mêmes ne doivent pas nécessairement correspondre, mais la passerelle doit disposer d'une couverture de sous-réseau dans chaque zone de disponibilité utilisée par le service.

Une passerelle qui ne répond pas à ces conditions est exclue. Si au moins une passerelle compatible est disponible, le service s'y connecte. Si aucune passerelle compatible n'est disponible, la demande

échoue. Rapports Amazon ECS `has associated gateways but none are in the service's VPC` lorsqu'aucune passerelle associée ne se trouve dans le VPC du service, et `none of the gateways associated with cluster match` lorsque le filtre de type d'accès exclut toutes les passerelles restantes.

Pour identifier et résoudre un problème de compatibilité

1. Vérifiez le filtre de type d'accès du service. Vérifiez que `filters.accessTypes` correspond au type d'accès de la passerelle à laquelle vous souhaitez vous connecter.
2. Récupérez les sous-réseaux configurés du service. Utiliser `list-service-deployments` pour identifier l'ARN de révision de service pertinent.

   ```
   aws ecs list-service-deployments \
       --cluster my-cluster \
       --service my-service \
       --query 'serviceDeployments[].targetServiceRevisionArn'
   ```

   Ensuite, utilisez `describe-service-revisions` pour récupérer ses sous-réseaux configurés.

   ```
   aws ecs describe-service-revisions \
       --service-revision-arns <service-revision-arn> \
       --query 'serviceRevisions[0].networkConfiguration.awsvpcConfiguration.subnets'
   ```

3. Récupérez les sous-réseaux configurés de la passerelle. Utiliser `describe-gateway` pour afficher la configuration réseau de la passerelle.

   ```
   aws ecs describe-gateway \
       --gateway my-gateway
   ```

4. Comparez les VPC et les zones de disponibilité. Utiliser `describe-subnets` pour récupérer l'ID VPC et la zone de disponibilité pour les sous-réseaux de service et de passerelle.

   ```
   aws ec2 describe-subnets \
       --subnet-ids <subnet-ids> \
       --query 'Subnets[].
   {SubnetId:SubnetId,VpcId:VpcId,AvailabilityZone:AvailabilityZone}'
   ```

   Vérifiez que le service et la passerelle se trouvent dans le même VPC et que la passerelle couvre toutes les zones de disponibilité utilisées par le service.

5. Résolvez le décalage. Mettez à jour le filtre de type d'accès ou la configuration de sous-réseau du service, le cas échéant. Si aucune passerelle compatible n'est disponible, contactez l'équipe de votre plateforme pour configurer ou associer une passerelle qui correspond au type d'accès, au VPC et aux exigences de zone de disponibilité du service.

Une fois l'incompatibilité résolue, réessayez de créer ou de mettre à jour le service.

> **Note**
>
> Pour une plus grande flexibilité lors de la connexion des services ECS, configurez des passerelles avec des sous-réseaux qui couvrent toutes les zones de disponibilité dans lesquelles vos services sont susceptibles d'être exécutés. Une couverture plus étendue de la zone de disponibilité permet à une passerelle de prendre en charge un plus grand nombre de services et réduit la nécessité de créer des passerelles supplémentaires. Les sous-réseaux de passerelle ne peuvent pas être modifiés après leur création.

Le connecteur Gateway fait référence à un port de conteneur non valide

Un connecteur de passerelle `portName` doit correspondre au nom d'un mappage de port dans la définition de tâche du service. Il s'agit du nom du port, et non du port de conteneur numérique. Lorsqu'aucun conteneur de la définition de tâche ne possède de mappage de port portant ce nom, Amazon ECS rapporte `InvalidConfiguration: "portName not found in task definition portMappings"`. Pour inspecter les mappages de ports nommés, exécutez :

```
aws ecs describe-task-definition \
    --task-definition my-task-definition:1 \
    --query 'taskDefinition.containerDefinitions[].portMappings[].name'
```

Corrigez le connecteur ou la définition de tâche pour que les noms correspondent, puis recommencez l'opération de service. Vérifiez cette option après avoir mis à jour une définition de tâche, car un nom de port précédemment valide n'existe peut-être plus.

La mise à jour des bilans de santé échoue avec la stratégie de déploiement ROLLING

La mise à jour des paramètres de vérification de l'état d'un connecteur de passerelle renvoie une erreur indiquant que les propriétés du groupe cible ne peuvent pas être modifiées avec `ROLLING` stratégie de déploiement. La modification de ces paramètres nécessite un groupe cible de

remplacement et un service utilisant le `ROLLING` la stratégie de déploiement ne prend pas en charge ce remplacement lors d'une mise à jour.

Conservez les paramètres de contrôle de santé existants ou utilisez un blue/green ou stratégie de déploiement de Canary pour le changement. Si vous devez déconnecter puis reconnecter le service pour remplacer le groupe cible, prévoyez de supprimer et de recréer son itinéraire de passerelle et son nom d'hôte. Pour la configuration des connecteurs, voir the section called “Connectez les services à une passerelle”.

Défaillances de provisionnement des itinéraires de passerelle

Une connexion de service peut avoir une passerelle dans `CREATE_FAILED` ou `UPDATE_FAILED` état, ou un déploiement de service peut ne pas terminer l'entrée de provisionnement. Une défaillance au niveau de l'itinéraire ne modifie pas nécessairement l'état général de la passerelle ni n'affecte l'itinéraire existant d'un autre service.

1. Trouvez les routes de passerelle défaillantes.

```
aws ecs list-gateway-routes \
    --gateway my-gateway \
    --status CREATE_FAILED UPDATE_FAILED
```

2. Identifiez le service auquel appartient chaque itinéraire défaillant, ainsi que les ressources que la passerelle a créées pour celui-ci.

```
aws ecs describe-gateway-route \
    --gateway-route-arn arn:aws:ecs:us-west-2:123456789012:gateway-route/my-gateway/
a1b2c3d4/9f8e7d6c
```

3. Vérifiez l'état de déploiement du service et les événements du service pour déterminer la raison de l'échec. Déterminez à partir de la raison de l'échec de l'émission d'un certificat, d'une règle d'écoute, d'un groupe cible, d'un groupe de sécurité ou d'une autre ressource.
4. Corrigez le problème sous-jacent et réessayez de déployer le service concerné. Une modification de la configuration de la passerelle ne relance pas nécessairement un déploiement de service qui a déjà échoué.

> **Note**
>
> Un itinéraire de passerelle signale un état de défaillance mais n'en indique pas la raison. Amazon ECS enregistre la raison du déploiement du service qui a déclenché le provisionnement, ainsi que dans les événements du service.

Les causes courantes incluent l'insuffisance des autorisations de rôle dans l'infrastructure, la capacité de l'équilibreur de charge ou les quotas de service, l'épuisement des adresses IP des sous-réseaux et la suppression ou la modification de ressources en dehors d'Amazon ECS. Vous pouvez y voir deux raisons : `InsufficientAvailabilityZones`, ce qui signifie que les sous-réseaux de la passerelle se trouvent dans moins de deux zones de disponibilité, le minimum pour un équilibreur de charge d'application, et `InvalidSubnet`, ce qui signifie qu'aucun sous-réseau configuré n'existe, appartient à un autre VPC ou ne possède plus d'adresse IP libre. Les sections qui suivent décrivent comment étudier les conditions d'autorisation, de capacité et hors bande. Pour les procédures de surveillance, voir the section called “Surveiller une passerelle”.

Pour les autorisations, `hostSharedAlb` type de passerelle requis, voir AmazonECSInfrastructureRoleForGatewayHostSharedALB dans le AWS Référence des politiques gérées.

Problèmes de capacité et de quotas

Pour les valeurs des quotas, voir the section called “Limites du service Gateway”.

Le cluster a atteint sa limite d'association de passerelles

Un cluster peut être associé à au plus 5 passerelles. Lorsqu'il atteint cette limite, Amazon ECS rapporte `Cluster has reached the maximum number of gateway associations (5)`. Dissociez une passerelle dont le cluster n'a plus besoin, ce qui nécessite de déconnecter d'abord les services qui l'utilisent encore.

La passerelle a atteint son nombre maximum d'équilibreurs de charge

Lorsque les équilibreurs de charge existants d'une passerelle ne peuvent pas accepter une autre connexion de service, Amazon ECS met en service un autre équilibreur de charge jusqu'à ce qu'il atteigne celui de la passerelle `maxAlbCount` réglage, dont la valeur par défaut est 10. Un équilibreur de charge peut fournir jusqu'à 25 services ECS. Lorsque chaque équilibreur de charge est plein

et que la passerelle est `maxAlbCount`, le déploiement d'un nouveau service échoue. Vérifiez le `currentAlbCount` contre `maxAlbCount` et inspectez les ressources gérées :

```
aws ecs describe-gateway --gateway my-gateway
aws ecs describe-gateway-resources --gateway my-gateway
```

Si `maxAlbCount` est le facteur limitant, augmentez-le dans la limite prise en charge. Par exemple :

```
aws ecs update-gateway \
    --gateway my-gateway \
    --configuration '{"hostSharedAlb":{"maxAlbCount":25}}'
```

Attendez que la mise à jour de la passerelle soit terminée, puis redéployez le service concerné. En hausse `maxAlbCount` ne réessaie pas automatiquement un déploiement de service qui a échoué.

Un sous-jacent AWS le quota de ressources a été atteint

Les ressources fournies par ECS Gateway sont soumises aux quotas de leurs propres services, y compris les équilibreurs de charge, les groupes cibles, les règles d'écoute, les certificats et les règles des groupes de sécurité. Le provisionnement peut également échouer si les sous-réseaux configurés ne disposent pas d'un nombre suffisant d'adresses IP disponibles. Vous pouvez voir les raisons suivantes :

- `LoadBalancerQuotaExceeded` et `TargetGroupQuotaExceeded`, pour connaître les limites du compte dans la Région.
- `ResourceQuotaExceeded`, pour une limite par équilibreur de charge, telle que des règles d'écoute ou des certificats.
- `CertificateQuotaExceeded`, pour la limite de certificats du compte.
- `CertificateAnnualLimitExceeded`, pour les certificats demandés au cours des 365 derniers jours. La suppression de certificats ne réduit pas ce nombre, donc celui-ci nécessite une augmentation du quota via AWS Soutien.
- `SecurityGroupCapacityExceeded` et `SecurityGroupRuleCapacityExceeded`, pour les groupes de sécurité par VPC et les règles par groupe de sécurité.
- `TestRuleLimitExceeded`, pour tester les règles de trafic sur un service.
- `TagLimitExceeded`, pour une combinaison de balises de passerelle et de balises de service dépassant les limites autorisées par une ressource, ou pour les balises rejetées par la politique des

balises de compte. Amazon ECS applique également ses propres `AmazonECSManaged` tag, qui utilise un emplacement.

Utilisez la raison de l'échec pour identifier la ressource limitée, puis vérifiez le quota de service applicable. Demandez une augmentation de quota, libérez la capacité inutilisée ou utilisez une autre configuration prise en charge, puis réessayez le déploiement du service. Pour les limites d'Elastic Load Balancing, voir Quotas pour les équilibreurs de charge de vos applications.

Problèmes liés à l'IAM et aux autorisations

Les opérations de passerelle impliquent deux identités différentes : le principal IAM qui effectue la demande d'API et le rôle IAM de l'infrastructure qu'Amazon ECS assume pour gérer AWS ressources. Un problème d'autorisation sur l'un n'est pas résolu en modifiant l'autre.

L'appelant ne peut pas passer le rôle d'infrastructure

`CreateGateway` et `UpdateGateway` échouer avec une erreur d'autorisation impliquant `iam:PassRole` lorsque l'appelant ne peut pas passer le rôle fourni dans `infrastructureRoleArn`. Il s'agit de l'échec de première exécution le plus courant, car l'appel échoue même lorsque l'appelant est autorisé à créer des passerelles. Vérifiez que l'appelant a `iam:PassRole` autorisation sur le rôle exact de l'ARN dans `infrastructureRoleArn`, et vérifiez les limites des autorisations, les politiques de contrôle des services et les refus explicites. L'ajout d'autorisations au rôle d'infrastructure ne résout pas une autorisation manquante sur l'identité de l'appelant. Pour la configuration des rôles IAM, voir the section called “Rôle IAM d'infrastructure pour Amazon ECS Gateway”.

Amazon ECS ne peut pas assumer le rôle d'infrastructure

Amazon ECS pouvait accéder à ce rôle mais ne pouvait pas l'assumer. Vérifiez que le rôle IAM existe et que sa politique de confiance le permet `ecs.amazonaws.com` appeler `sts:AssumeRole`.

```
aws iam get-role --role-name ecsInfrastructureRoleForGateways
```

La politique de confiance du rôle doit correspondre à ce qui suit.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
```

```
            "Sid": "AllowAccessInfrastructureForECSGateways",
            "Effect": "Allow",
            "Principal": {
                "Service": "ecs.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Si le rôle vient d'être créé ou modifié, attendez que la modification d'IAM se propage avant de réessayer. Une relation de confiance manquante ou incorrecte n'est pas corrigée en accordant à l'appelant `iam:PassRole` seul. Vérifiez également que le rôle se trouve dans le même compte et la même partition que la passerelle. Un échec connexe, `AWS STS is not activated for this account in a Region`, signifie AWS Security Token Service n'est pas activé pour cette région. Activez-le, puis réessayez.

Le rôle d'infrastructure ne peut pas gérer une ressource

Un itinéraire de passerelle ou une mise à jour de passerelle signale une erreur d'accès refusé pour un Elastic Load Balancing, Amazon EC2 ou AWS Certificate Manager opération lorsque le rôle d'infrastructure n'a pas d'autorisation. Vérifiez que le rôle dispose des autorisations requises par la configuration de la passerelle. Pour `hostSharedAlb` type de passerelle, vérifiez la pièce jointe AWS politique gérée.

```
aws iam list-attached-role-policies --role-name ecsInfrastructureRoleForGateways
```

Joignez le `AmazonECSInfrastructureRoleForGatewayHostSharedALB` stratégie gérée ou, si vous l'avez remplacée par votre propre stratégie, comparez cette politique à l'action IAM nommée lors de l'échec. Vous pouvez consulter les autorisations accordées par cette politique dans le AWS Référence des politiques gérées. Vérifiez également si une limite d'autorisations ou une autre restriction de compte empêche l'action. L'opération échouée exacte et l'ARN de la ressource sont plus utiles qu'un code indifférencié `AccessDenied` message.

Connectivité des services et santé de la cible

Le nom d'hôte du service ne répond pas

Lorsque les demandes adressées à un nom d'hôte HTTPS généré par la passerelle expirent ou échouent, vérifiez la connexion entre la route de la passerelle et l'application.

1. Vérifiez l'état de l'itinéraire de la passerelle. Trouvez l'itinéraire pour le service et confirmez-le `ACTIVE`. Un itinéraire dans `CREATE_IN_PROGRESS` n'a pas terminé l'approvisionnement, et un itinéraire dans un `*_FAILED` l'État a un échec de provisionnement. Inspectez la défaillance de l'itinéraire avant de dépanner l'application :

   ```
   aws ecs list-gateway-routes --cluster my-cluster --service my-service
   ```

2. Vérifiez l'accès à la passerelle. Une passerelle interne nécessite une connectivité réseau à l'équilibreur de charge interne. Vérifiez que vous effectuez le test à partir d'un réseau qui peut y accéder.

3. Vérifiez l'état de santé de la cible. Obtenez l'ARN du groupe cible à partir de la connexion de passerelle du service, qui `describe-service-revisions` renvoie, puis inspecte ses cibles enregistrées.

   ```
   aws elbv2 describe-target-health \
       --target-group-arn arn:aws:elasticloadbalancing:us-
   west-2:123456789012:targetgroup/my-service/0123456789abcdef
   ```

   Un groupe cible peut signaler des cibles défectueuses lorsque l'application renvoie un code d'état inattendu, expire, écoute sur un port différent ou ne peut pas être atteinte par l'équilibreur de charge. La réponse inclut l'état de santé de chaque cible et, le cas échéant, une raison et une description. Par exemple, une raison de `Target.ResponseCodeMismatch` indique que la réponse ne correspond pas à la plage de codes de réussite configurée. Vérifiez le chemin du contrôle de santé, la réponse de l'application et le port cible avant de modifier une infrastructure de passerelle. Pour les codes des motifs du bilan de santé, voir Contrôles de santé pour les groupes cibles d'Application Load Balancer.

   Un `ACTIVE` la route de passerelle signifie que le provisionnement d'entrée a réussi. Cela ne signifie pas que les cibles de l'application sont saines.

4. Vérifiez les groupes de sécurité. Vérifiez que le groupe de sécurité entrant de l'équilibreur de charge autorise le trafic en provenance des clients prévus et que le groupe de sécurité des tâches autorise le trafic provenant de l'équilibreur de charge sur le port de conteneur. Lorsque l'équipe de la plateforme fournit le groupe de sécurité entrant de la passerelle, Amazon ECS ne gère pas ses règles entrantes.

5. Vérifiez l'application et les commandes de trafic. Vérifiez le port d'écoute et le point final d'intégrité du conteneur. Si un AWS WAF l'ACL web est configurée, vérifiez si une règle bloque la demande.

Utilisez les journaux d'accès de l'équilibreur de charge et CloudWatch des mesures permettant de distinguer les erreurs de l'équilibreur de charge des réponses des applications.

Échecs de mise à jour

`DescribeGateway` rapports `UPDATE_FAILED` après une mise à jour de configuration de passerelle qu'Amazon ECS n'a pas pu appliquer à tous les équilibreurs de charge. Amazon ECS applique une mise à jour à chaque équilibreur de charge géré à tour de rôle et fait de son mieux pour tous les mettre à jour plutôt que de s'arrêter au premier échec. Certains équilibreurs de charge peuvent donc accepter la modification tandis que d'autres échouent. Ne partez pas du principe que la mise à jour a été annulée sur la passerelle. Récupérez l'état et les informations de panne.

```
aws ecs describe-gateway --gateway my-gateway
```

Inspectez `statusReason` et `failures` pour les ressources et les opérations qui n'ont pas pu être mises à jour. Par exemple, une erreur concernant un identifiant de groupe de sécurité non valide pointe vers le groupe de sécurité entrant configuré pour la passerelle, et une erreur d'accès refusé pointe vers les autorisations disponibles pour le rôle d'infrastructure. Pour savoir comment lire le `failures` liste, voir the section called “Localisez les détails des défaillances”. Corrigez la cause signalée et réessayez la mise à jour demandée. Attendez que la passerelle signale une mise à jour réussie avant de supposer que chaque équilibreur de charge géré possède la nouvelle configuration, et attendez que la mise à jour soit terminée avant de réessayer.

> **Important**
>
> Une mise à jour partielle de la passerelle peut laisser différents équilibreurs de charge dotés de configurations différentes, ce qui peut modifier le comportement du trafic ou des politiques pour les services qui les sous-tendent. Traitez la mise à jour comme partiellement appliquée jusqu'à ce que la passerelle fasse rapport `UPDATE_COMPLETE`, et validez les ressources concernées après la restauration.

Problèmes de dissociation et de suppression

Une association de clusters reste DISSOCIANTE

Une association de clusters de passerelles reste active `DISASSOCIATING` jusqu'à ce que toutes les routes de passerelle associées à ce cluster soient supprimées. Amazon ECS ne redéploie pas

automatiquement les services connectés et ne supprime pas leurs itinéraires existants lorsque vous dissociez une passerelle.

Pour identifier les services toujours connectés, utilisez la commande suivante :

```
aws ecs list-gateway-routes --gateway my-gateway --cluster my-cluster
```

Coordonnez-vous avec les équipes chargées de l'application pour mettre à jour ou redéployer les services concernés afin qu'ils se déconnectent de la passerelle. Lorsque le dernier itinéraire est supprimé, la dissociation est terminée.

La passerelle ne peut pas être supprimée

Une passerelle ne peut pas être supprimée tant que la protection contre la suppression est activée ou lorsqu'elle est associée à des clusters dans `ACTIVE` ou `DISASSOCIATING` état. Vérifiez les associations de clusters de la passerelle et assurez-vous que toutes les routes de passerelle restantes sont supprimées afin que la dissociation puisse être terminée. Une fois toutes les associations supprimées, assurez-vous que la protection contre la suppression est désactivée, puis réessayez de supprimer la passerelle. Pour la procédure complète, voir the section called “Supprimer une passerelle”.

```
aws ecs update-gateway --gateway my-gateway --no-deletion-protection
aws ecs delete-gateway --gateway my-gateway
```

Modifications apportées en dehors d'Amazon ECS

Amazon ECS ne répare ni ne recrée automatiquement les ressources gérées par la passerelle qui sont modifiées ou supprimées en dehors d'Amazon ECS. La modification ou la suppression de ressources gérées telles que des équilibreurs de charge, des écouteurs, des règles d'écoute, des groupes cibles, des certificats ou des groupes de sécurité extérieurs à Amazon ECS peut interrompre le trafic ou entraîner l'échec de déploiements de services ultérieurs, de mises à jour de passerelles ou d'opérations de nettoyage.

Si une ressource gérée par passerelle a été modifiée ou supprimée en dehors d'Amazon ECS, procédez comme suit :

1. Identifier la ressource concernée: Utilisation `describe-gateway-resources` ou `describe-gateway-route` pour identifier les ressources gérées. Inspectez leur configuration actuelle dans

le AWS service permettant d'identifier les ressources manquantes ou modifiées. Une modification apportée en dehors d'Amazon ECS peut apparaître comme l'une des raisons d'échec suivantes :

- `ManagedResourceDeleted`, lorsqu'une ressource créée par Amazon ECS pour le service a été supprimée en dehors d'Amazon ECS.
- `ConflictingListener`, lorsqu'un écouteur existe sur le port de la passerelle sur l'équilibreur de charge et qu'Amazon ECS ne l'a pas créé.
- `SecurityGroupNameInUse`, lorsqu'un groupe de sécurité portant le nom souhaité par Amazon ECS existe et n'est pas géré par Amazon ECS. Amazon ECS ne le reconfigure ni ne le supprime.
- `InvalidSecurityGroup`, lorsque le groupe de sécurité entrant de la passerelle n'existe plus.

2. Restaurez le service concerné: si une ressource a été modifiée, restaurez sa configuration d'origine et redéployez le service. Si une ressource a été supprimée, supprimez le connecteur de passerelle du service et déployez-le, puis ajoutez-le à nouveau et déployez-le à nouveau. Le second déploiement fournit de nouvelles ressources et une nouvelle passerelle. Étant donné que le service reçoit un nouveau nom d'hôte généré, mettez à jour tous les clients ou enregistrements DNS qui pointent vers l'ancien.
3. Vérifiez la connectivité: confirmez que la route de passerelle du service est `ACTIVE` et ses cibles sont saines avant que vous ne redirigiez le trafic vers ce site.

> **Important**
>
> Ne modifiez ni ne supprimez `AmazonECSManaged` balise provenant de ressources gérées par la passerelle. Amazon ECS l'utilise pour identifier les ressources qu'il gère, sans quoi Amazon ECS ne peut plus mettre à jour ni nettoyer ces ressources. Apportez les modifications de configuration prises en charge via les opérations d'Amazon ECS Gateway plutôt que de modifier directement les ressources sous-jacentes.

Définitions de tâche Amazon ECS

Une définition de tâche est le plan de votre application. Il s'agit d'un fichier texte au format JSON qui décrit les paramètres et un ou plusieurs conteneurs qui forment votre application.

Voici certains des paramètres que vous pouvez spécifier dans une définition de tâche :

- La capacité à utiliser, qui détermine l’infrastructure sur laquelle vos tâches sont hébergées
- L'image Docker à utiliser avec chaque conteneur dans la tâche
- Les ressources d'UC et de mémoire à utiliser avec chaque tâche ou chaque conteneur au sein d'une tâche
- Les exigences en matière de mémoire et d’UC
- Le système d’exploitation du conteneur sur lequel la tâche s’exécute
- Le mode réseau Docker à utiliser pour les conteneurs dans votre tâche
- La configuration de journalisation à utiliser pour les tâches
- Si la tâche continue à s'exécuter en cas d'arrêt ou d'échec du conteneur
- La commande que le conteneur exécute au démarrage
- Les volumes de données utilisés avec les conteneurs dans la tâche
- Le rôle IAM que les tâches utilisent

Pour obtenir la liste complète des paramètres de définition de tâche, veuillez consulter Paramètres de définition de tâche Amazon ECS pour Fargate.

Après avoir créé une définition de tâche, vous pouvez l'exécuter en tant que tâche ou service.

- Une tâche est l'instanciation d'une définition de tâche au sein d'un cluster. Après avoir créé une définition de tâche pour votre application dans Amazon ECS, vous pouvez spécifier le nombre de tâches à exécuter sur votre cluster.
- Un service Amazon ECS service exécute et conserve simultanément le nombre souhaité de tâches dans un cluster Amazon ECS. Le principe est le suivant : si l'une de vos tâches échoue ou s'arrête pour une raison quelconque, le planificateur de service d'Amazon ECS service lance une autre instance en fonction de votre définition de tâche. Il procède ainsi pour le remplacer et donc maintenir le nombre de tâches souhaité dans le service.

Rubriques

- États de définition de tâche Amazon ECS
- Architecture de votre application pour Amazon ECS
- Création d'une définition de tâche Amazon ECS à l'aide de la console
- Utilisation d'Amazon Q Developer pour proposer des recommandations de définition de tâche dans la console Amazon ECS
- Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console
- Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console
- Suppression de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console
- Cas d'utilisation de définition de tâche Amazon ECS
- Paramètres de définition de tâche Amazon ECS pour les instances gérées Amazon ECS
- Paramètres de définition de tâche Amazon ECS pour Fargate
- Paramètres de définition de tâche Amazon ECS pour Amazon EC2
- Modèle de définition de tâche Amazon ECS
- Exemple de définitions de tâches Amazon ECS

États de définition de tâche Amazon ECS

La définition d'une tâche change d'état lorsque vous la créez, annulez son enregistrement ou la supprimez. Vous pouvez afficher l'état de la définition de tâche dans la console ou en utilisant `DescribeTaskDefinition`.

Les états possibles pour une définition de tâche sont les suivants :

ACTIF

La définition d'une tâche est `ACTIVE` après son enregistrement auprès d'Amazon ECS. Les définitions de tâches à l'état `ACTIVE` vous permettent d'exécuter des tâches ou de créer des services.

INACTIVE

Une définition de tâche passe de l'état `ACTIVE` à l'état `INACTIVE` lorsque vous annulez l'enregistrement d'une définition de tâche. Vous pouvez récupérer la définition d'une tâche

`INACTIVE` en appelant `DescribeTaskDefinition`. Vous ne pouvez pas exécuter de nouvelles tâches ou créer de nouveaux services avec une définition de tâche à l'état `INACTIVE`. Il n'y a aucun impact sur les services ou les tâches existants.

DELETE_IN_PROGRESS

Une définition de tâche passe de l'état `INACTIVE` à l'état `DELETE_IN_PROGRESS` une fois que vous l'avez soumise pour suppression. Une fois que la définition de tâche est à l'état `DELETE_IN_PROGRESS`, Amazon ECS vérifie régulièrement que la définition de tâche cible n'est référencée par aucune tâche ou déploiement actif, puis supprime définitivement la définition de tâche. Vous ne pouvez pas exécuter de nouvelles tâches ou créer de nouveaux services avec une définition de tâche à l'état `DELETE_IN_PROGRESS`. Une définition de tâche peut être soumise pour suppression à tout moment sans que cela n'ait d'impact sur les tâches et les services existants.

Les définitions de tâches dont l'état est `DELETE_IN_PROGRESS` peuvent être consultées dans la console et vous pouvez récupérer la définition de tâche en appelant `DescribeTaskDefinition`.

Lorsque vous supprimez toutes les révisions de définition de tâche `INACTIVE`, le nom de la définition de tâche n'est pas affiché dans la console et n'est pas renvoyé dans l'API. Si une révision de définition de tâche est à l'état `DELETE_IN_PROGRESS`, le nom de la définition de tâche est affiché dans la console et renvoyé dans l'API. Le nom de la définition de tâche est conservé par Amazon ECS et la révision est incrémentée la prochaine fois que vous créerez une définition de tâche portant ce nom.

Si vous utilisez AWS Config pour gérer vos définitions de tâches, AWS Config vous facture tous les enregistrements de définitions de tâches. Vous n'êtes facturé que pour la désinscription de la dernière définition de tâche `ACTIVE`. La suppression d'une définition de tâche est gratuite. Pour plus d'informations sur la tarification, consultez Tarification d'AWS Config.

Ressources Amazon ECS pouvant bloquer une suppression

Aucune requête de suppression de définition de tâche ne sera traitée tant qu'il existe des ressources Amazon ECS qui dépendent de la révision de la définition de tâche. Les ressources suivantes peuvent empêcher la suppression d'une définition de tâche :

- Tâches autonomes Amazon ECS : la définition de tâche est nécessaire pour que la tâche reste saine.

- Tâches d’un service Amazon ECS : la définition de tâche est nécessaire pour que la tâche reste saine.
- Déploiements et jeux de tâches d’un service Amazon ECS : la définition de tâche est nécessaire lorsqu’un événement de mise à l’échelle est lancé pour un déploiement ou un jeu de tâches Amazon ECS.

Si votre définition de tâche reste dans le `DELETE_IN_PROGRESS` état, vous pouvez utiliser la console ou le AWS CLI pour identifier, puis arrêter les ressources qui bloquent la suppression de la définition de tâche.

Suppression de la définition de tâche après la suppression de la ressource bloquée

Les règles suivantes s'appliquent une fois que vous avez supprimé les ressources bloquant la suppression de la définition de tâche :

- Tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à une heure après l'arrêt de la tâche.
- Déploiements et jeux de tâches d’un service Amazon ECS : la suppression de la définition de tâche peut prendre jusqu’à 24 heures après la suppression du déploiement ou du jeu de tâches.

Architecture de votre application pour Amazon ECS

Vous élaborez votre application en créant une définition de tâche pour votre application. La définition de tâche contient les paramètres qui définissent les informations relatives à l’application, notamment :

- La capacité à utiliser, qui détermine l’infrastructure sur laquelle vos tâches sont hébergées.

  Lorsque vous utilisez le fournisseur de capacité EC2, vous choisissez également le type d’instance. Lorsque vous utilisez le fournisseur de capacité d’instances gérées Amazon ECS, vous pouvez définir les exigences relatives aux instances pour qu’Amazon ECS gère la capacité de calcul. Pour certains types d’instances, tels que le GPU, vous devez définir des paramètres spécifiques. Pour plus d’informations, consultez Cas d’utilisation de définition de tâche Amazon ECS.
- L’image de conteneur, qui contient le code de votre application et toutes les dépendances nécessaires à son exécution.
- Le mode réseau à utiliser pour les conteneurs dans votre tâche.

  Le mode réseau détermine la manière dont votre tâche communique sur le réseau.

Pour les tâches exécutées sur des instances EC2 et des instances gérées Amazon ECS, plusieurs options sont disponibles, mais nous vous recommandons d'utiliser le mode réseau `awsvpc`. Le mode réseau `awsvpc` simplifie la mise en réseau des conteneurs en vous offrant davantage de contrôle sur la manière dont vos applications communiquent entre elles et avec d'autres services au sein de vos VPC.

Pour les tâches exécutées sur Fargate, vous devez utiliser le mode réseau `awsvpc`.

- La configuration de journalisation à utiliser pour les tâches
- Les volumes de données utilisés avec les conteneurs dans la tâche.

Pour obtenir la liste complète des paramètres de définition de tâche, veuillez consulter Paramètres de définition de tâche Amazon ECS pour Fargate.

Suivez ces recommandations lors de la création de vos définitions de tâche :

- Utilisez chaque famille de définitions de tâches pour un seul objectif métier.

  Si vous regroupez plusieurs types de conteneurs d'applications dans la même définition de tâche, vous ne pouvez pas mettre à l'échelle ces conteneurs indépendamment. Par exemple, un site Web et une API nécessitent généralement des modèles de mise à l'échelle différents. À mesure que le trafic augmente, le nombre de conteneurs Web dont vous aurez besoin pourra différer de celui des conteneurs API. Si ces deux conteneurs sont déployés dans la même définition de tâche, chaque tâche exécute le même nombre de conteneurs Web et de conteneurs d'API.
- Associez chaque version d'application à une révision de définition de tâche au sein d'une famille de définitions de tâches.

  Au sein d'une famille de définitions de tâches, chaque révision de définition de tâche représente un instantané ponctuel des paramètres d'une image de conteneur particulière. De la même manière, le conteneur est un instantané de tous les composants nécessaires à l'exécution d'une version particulière du code de votre application.

  Créez un mappage un-à-un entre une version du code d'application, une balise d'image de conteneur et une révision de définition de tâche. Un processus de publication typique implique une validation git qui est transformée en une image de conteneur balisée avec le SHA de validation git. Cette balise d'image de conteneur reçoit ensuite sa propre révision de définition de tâche Amazon ECS. Enfin, le service Amazon ECS est mis à jour afin de déployer la nouvelle révision de la définition de tâche.

- Utilisez différents rôles IAM pour chaque famille de définitions de tâches.

  Définissez chaque définition de tâche avec son propre rôle IAM. Mettez cette pratique en œuvre tout en fournissant à chaque composante métier sa propre famille de définitions de tâches. En mettant en œuvre ces deux bonnes pratiques, vous pouvez limiter l'accès de chaque service aux ressources de votre AWS compte. Par exemple, vous pouvez autoriser votre service d'authentification à accéder à votre base de données de mots de passe. Dans le même temps, vous pouvez vous assurer que seul votre service de commande a accès aux informations de paiement par carte de crédit.

Bonnes pratique en matière de taille des tâches Amazon ECS

La taille de vos conteneurs et de vos tâches est essentielle pour la mise à l'échelle et la planification de la capacité. Dans Amazon ECS, l'UC et la mémoire sont deux indicateurs de ressources utilisés pour déterminer la capacité. Le processeur est mesuré en unités de 1/1024 d'un processeur virtuel complet (où 1 024 unités équivalent à 1 processeur virtuel entier). La mémoire est mesurée en mébioctets. Dans la définition de votre tâche, vous pouvez configurer les réserves et les limites des ressources.

Lorsque vous configurez une réserve, vous définissez la quantité minimale de ressources requise par une tâche. Votre tâche reçoit au minimum la quantité de ressources demandée. Votre application pourrait être en mesure d'utiliser une quantité d'UC ou de mémoire supérieure à la réserve que vous avez déclarée. Toutefois, les limites que vous avez également déclarées s'appliquent. L'utilisation d'une quantité supérieure à la réserve est appelée éclatement (bursting). Dans Amazon ECS, les réserves sont garanties. Par exemple, si vous utilisez des instances Amazon EC2 pour fournir de la capacité, Amazon ECS ne place aucune tâche sur une instance où la réserve ne peut pas être honorée.

Une limite est la quantité maximale d'unités d'UC ou de mémoire que votre conteneur ou votre tâche peut utiliser. Toute tentative d'utiliser davantage d'UC que cette limite entraîne un ralentissement. Toute tentative d'utiliser davantage de mémoire entraîne l'arrêt de votre conteneur.

Choisir ces valeurs peut s'avérer difficile. En effet, les valeurs les mieux adaptées à votre application dépendent dans une large mesure des besoins en ressources de celle-ci. Le test de charge de votre application est la clé d'une planification réussie des besoins en ressources et d'une meilleure compréhension des exigences de votre application.

Applications sans état

Pour les applications sans état soumises à une mise à l'échelle horizontale, telles que les applications derrière un équilibreur de charge, nous vous recommandons de déterminer au préalable la quantité de mémoire consommée par votre application lorsqu'elle traite les requêtes. Pour cela, vous pouvez utiliser des outils traditionnels tels que `ps` ou `top`, ou des solutions de surveillance telles que CloudWatch Informations sur les conteneurs.

Lorsque vous déterminez une réserve d'UC, considérez de quelle manière votre application doit être mise à l'échelle pour répondre aux besoins de votre entreprise. Vous pouvez utiliser des réservations de processeurs plus petites, telles que 256 unités de processeur (ou 1/4 vCPU), pour évoluer de manière précise tout en minimisant les coûts. Cependant, leur mise à l'échelle peut ne pas être assez rapide pour répondre à des pics de demande importants. Vous pouvez utiliser des réserves d'UC plus importantes pour procéder plus rapidement à des augmentations ou à des réductions horizontales et ainsi répondre plus rapidement aux pics de demande. Cependant, les réserves d'UC plus importantes sont plus coûteuses.

Autres applications

Pour les applications qui ne subissent pas de mise à l'échelle horizontale, telles que les travailleurs uniques ou les serveurs de base de données, la capacité disponible et les coûts constituent vos considérations les plus importantes. Vous devez choisir la quantité de mémoire et d'UC en fonction des résultats des tests de charge, qui indiquent les ressources nécessaires pour traiter le trafic et atteindre votre objectif de niveau de service. Amazon ECS garantit que l'application est placée sur un hôte doté d'une capacité adéquate.

Mise en réseau des tâches Amazon ECS pour les instances gérées Amazon ECS

Le comportement de mise en réseau des tâches Amazon ECS exécutées sur des instances gérées Amazon ECS est déterminé par le mode réseau spécifié dans la définition de tâche. Vous devez spécifier un mode réseau dans la définition de tâche. Vous ne pourrez exécuter aucune tâche sur les instances gérées Amazon ECS en utilisant une définition de tâche qui ne spécifie pas de mode réseau. Les instances gérées Amazon ECS prennent en charge les modes réseau suivants, garantissant ainsi la rétrocompatibilité pour la migration de charges de travail depuis Fargate ou Amazon ECS vers Amazon EC2 :

Mode réseau	Description
`awsvpc`	Chaque tâche reçoit sa propre interface réseau Elastic (ENI) et son adresse IPv4 privée. Ce mode offre les mêmes propriétés réseau que les instances Amazon EC2 et est compatible avec les tâches Fargate traditionnelles. Utilise la jonction ENI pour une densité de tâches élevée.
`host`	Les tâches partagent directement l'espace de noms réseau de l'hôte. Le réseau de conteneurs est lié à l'instance hôte sous-jacente.

Utilisation d'un VPC dans IPv6-only mode

Dans un IPv6-only configuration, vos tâches Amazon ECS communiquent exclusivement via IPv6. Pour configurer des VPC et des sous-réseaux pour un IPv6-only configuration, vous devez ajouter

un bloc d'adresse CIDR IPv6 au VPC et créer des sous-réseaux qui incluent uniquement un bloc d'adresse CIDR IPv6. Pour plus d'informations, consultez les sections Ajout du support IPv6 à votre VPC et Création d'un sous-réseau dans le Guide de l'utilisateur Amazon VPC. Vous devez également mettre à jour les tables de routage avec des cibles IPv6 et configurer les groupes de sécurité avec des règles IPv6. Pour plus d'informations, consultez les sections Configuration des tables de routage et Configuration des règles des groupes de sécurité dans le Guide de l'utilisateur Amazon VPC.

Les considérations suivantes s'appliquent :

- Vous pouvez mettre à jour un IPv4-only ou un service Amazon ECS dualstack vers un IPv6-only configuration en mettant à jour directement le service pour l'utiliser IPv6-only sous-réseaux ou en créant un parallèle IPv6-only service et en utilisant les déploiements bleu-vert d'Amazon ECS pour transférer le trafic vers le nouveau service. Pour plus d'informations sur les déploiements bleu-vert Amazon ECS, consultez la section Amazon ECS blue/green déploiements.
- Un IPv6-only Le service Amazon ECS doit utiliser des équilibreurs de charge à double pile avec des groupes cibles IPv6. Si vous procédez à la migration d'un service Amazon ECS existant qui repose sur un Application Load Balancer ou un Network Load Balancer, vous pouvez créer un équilibreur de charge à double pile et lui transférer le trafic depuis l'ancien équilibreur de charge, ou mettre à jour le type d'adresse IP de l'équilibreur de charge existant.

  Pour plus d'informations sur les Network Load Balancer, consultez les sections Création d'un Network Load Balancer et Mise à jour des types d'adresses IP pour votre Network Load Balancer dans le Guide de l'utilisateur des Network Load Balancer. Pour plus d'informations sur les Application Load Balancer, consultez les sections Création d'un Application Load Balancer et Mise à jour des types d'adresses IP pour votre Application Load Balancer dans le Guide de l'utilisateur des Application Load Balancer.
- Pour les tâches Amazon ECS dans un IPv6-only configuration avec laquelle communiquer IPv4-only points de terminaison, que vous pouvez configurer DNS64 et NAT64 pour la traduction d'adresses réseau d'IPv6 vers IPv4. Pour plus d'informations, veuillez consulter la section DNS64 et NAT64 dans le Guide de l'utilisateur Amazon VPC.
- Charges de travail Amazon ECS dans un IPv6-only la configuration doit utiliser les points de terminaison URI des images Amazon ECR dualstack lors de l'extraction d'images depuis Amazon ECR. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

> **Note**
>
> Amazon ECR ne prend pas en charge les points de terminaison VPC à interface dualstack qui effectuent des tâches dans un IPv6-only la configuration peut utiliser. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

- Amazon ECS Exec n'est pas pris en charge dans un IPv6-only configuration.

Allocation d'une interface réseau pour les tâches sur les instances gérées Amazon ECS

L'utilisation du mode réseau `awsvpc` dans les instances gérées Amazon ECS simplifie la mise en réseau des conteneurs, car vous disposez d'un meilleur contrôle sur la manière dont vos applications communiquent entre elles et avec d'autres services au sein de vos VPC. Le mode réseau `awsvpc` fournit également une sécurité accrue pour vos conteneurs en vous permettant d'utiliser des groupes de sécurité et des outils de surveillance réseau à un niveau plus détaillé dans les tâches ECS.

Par défaut, chaque instance d'instances gérées Amazon ECS possède une interface réseau Elastic (ENI) de jonction attachée lors du lancement en tant qu'ENI principale lorsque le type d'instance prend en charge la jonction. Pour plus d'informations sur les types d'instances qui prennent en charge la jonction ENI, consultez la section Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.

> **Note**
>
> Lorsque le type d'instance choisi ne prend pas en charge les ENI de jonction, l'instance sera lancée avec une ENI standard.

Chaque tâche exécutée sur l'instance reçoit sa propre ENI attachée à l'ENI de jonction, avec une adresse IP privée principale. Si votre VPC est configuré pour le mode double pile et que vous utilisez un sous-réseau avec un bloc d'adresses CIDR IPv6, l'ENI reçoit également une adresse IPv6. Lorsque vous utilisez un sous-réseau public, vous pouvez éventuellement attribuer une adresse IP publique à l'ENI principale de l'instance gérée Amazon ECS en activant l'adressage public IPv4 pour le sous-réseau. Pour plus d'informations, consultez la section Modification des attributs d'adressage

IP de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC. Une tâche ne peut avoir qu'une seule ENI associée à la fois.

Les conteneurs qui appartiennent à la même tâche peuvent également communiquer via l'interface `localhost`. Pour plus d'informations sur les VPC et les sous-réseaux, consultez la section Fonctionnement d'Amazon VPC dans le Guide de l'utilisateur Amazon VPC.

Les opérations suivantes utilisent l'ENI principale attachée à l'instance :

- Téléchargements d'images : les images des conteneurs sont téléchargées depuis Amazon ECR via l'ENI principale.
- Récupération des secrets : les secrets et autres informations d'identification de Secrets Manager sont récupérés via l'ENI principale.
- Chargements de journaux - Les journaux sont téléchargés sur CloudWatch via l'ENI principal.
- Téléchargements de fichiers d'environnement : les fichiers d'environnement sont téléchargés via l'ENI principale.

Le trafic de l'application passe par l'ENI de tâche.

Dans la mesure où chaque tâche obtient sa propre ENI, vous pouvez utiliser des fonctionnalités de mise en réseau telles que les journaux de flux VPC, que vous pouvez utiliser pour surveiller le trafic vers et depuis vos tâches. Pour plus d'informations, consultez Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez également profiter de AWS PrivateLink. Vous pouvez configurer un point de terminaison d'interface VPC pour pouvoir accéder aux API d'Amazon ECS via des adresses IP privées. AWS PrivateLink limite l'ensemble du trafic réseau entre votre VPC et Amazon ECS sur le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle. Pour plus d'informations, consultez la section Points de terminaison de VPC d'interface Amazon ECS (AWS PrivateLink).

Le `awsvpc` le mode réseau vous permet également de tirer parti d'Amazon VPC Traffic Mirroring pour la sécurité et la surveillance du trafic réseau lorsque vous utilisez des types d'instance auxquels aucun ENI de jonction n'est connecté. Pour plus d'informations, consultez la section Qu'est-ce que la mise en miroir du trafic ? dans le Guide de mise en miroir du trafic Amazon VPC.

Considérations pour **`awsvpc`** mode

- Les tâches nécessitent le rôle lié au service Amazon ECS pour la gestion ENI. Ce rôle est automatiquement créé lorsque vous créez un cluster ou un service.
- Les ENI de tâche sont gérées par Amazon ECS et ne peuvent pas être détachées ou modifiées manuellement.
- L'attribution d'une adresse IP publique à l'ENI de tâche à l'aide de `assignPublicIp` lors de l'exécution d'une tâche autonome (`RunTask`) ou de la création ou de la mise à jour d'un service (`CreateService`/`UpdateService`) n'est pas prise en charge.
- Lorsque vous configurez le réseau `awsvpc` au niveau des tâches, vous devez utiliser le même VPC que celui que vous avez spécifié dans le modèle de lancement du fournisseur de capacité d'instances gérées Amazon ECS. Vous pouvez utiliser des sous-réseaux et des groupes de sécurité différents de ceux spécifiés dans le modèle de lancement.
- Pour les tâches en mode réseau `awsvpc`, utilisez le type de cible `ip` lors de la configuration des groupes cibles de l'équilibreur de charge. Amazon ECS gère automatiquement l'enregistrement des groupes cibles pour les modes réseau pris en charge.

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer via IPv4, IPv6 ou les deux. Les adresses IPv4 et IPv6 sont indépendantes les unes des autres. Vous devez donc configurer le routage et la sécurité dans votre VPC séparément pour IPv4 et IPv6. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la rubrique Migration vers IPv6 dans le Guide de l'utilisateur Amazon VPC.

Si vous avez configuré votre VPC avec une passerelle Internet ou une passerelle Internet sortante uniquement, vous pouvez utiliser votre VPC en mode double pile. De ce fait, les tâches auxquelles une adresse IPv6 est attribuée peuvent accéder à Internet via une passerelle Internet ou une passerelle Internet de sortie uniquement. Les passerelles NAT sont facultatives. Pour plus d'informations, voir Passerelles Internet et Egress-only passerelles Internet dans le Guide de l'utilisateur d'Amazon VPC.

Une adresse IPv6 est attribuée aux tâches Amazon ECS si les conditions suivantes sont réunies :

- L'instance gérée Amazon ECS qui héberge la tâche utilise la version `1.45.0` ou une version ultérieure de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de

l'agent utilisée par votre instance et la mise à jour si nécessaire, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.

- Le paramètre de compte `dualStackIPv6` est activé. Pour plus d'informations, consultez Accès aux fonctionnalités d'Amazon ECS avec les paramètres du compte.
- Votre tâche utilise le mode réseau `awsvpc`.
- Votre VPC et votre sous-réseau sont configurés pour IPv6. La configuration inclut les interfaces réseau créées dans le sous-réseau spécifié. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, veuillez consulter les rubriques Migration vers IPv6 et Modifier l'attribut d'adressage IPv6 de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Mode réseau hôte

En mode `host`, les tâches partagent directement l'espace de noms réseau de l'hôte. La configuration réseau du conteneur est liée à l'instance hôte des instances gérées Amazon ECS sous-jacente que vous spécifiez à l'aide du paramètre `networkConfiguration` lorsque vous créez un fournisseur de capacité d'instances gérées Amazon ECS.

L'utilisation de ce mode réseau présente des inconvénients importants. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait que seule la première tâche peut se lier au port requis sur l'instance Amazon EC2. Il est également impossible de remapper un port de conteneur lorsqu'il utilise le mode réseau `host`. Par exemple, si une application doit écouter un numéro de port spécifique, vous ne pouvez pas remapper le numéro de port directement. Vous devez plutôt gérer les conflits de ports en modifiant la configuration de l'application.

L'utilisation du mode réseau `host` a également des implications en termes de sécurité. Ce mode permet aux conteneurs de se faire passer pour l'hôte et de se connecter à des services réseau privés en boucle sur l'hôte.

N'utilisez le mode hôte que lorsque vous avez besoin d'un accès direct au réseau hôte ou lors de la migration d'applications nécessitant un accès réseau au niveau de l'hôte.

Options de mise en réseau des tâches Amazon ECS pour EC2

Le comportement de mise en réseau des tâches Amazon ECS hébergées sur des instances Amazon EC2 dépend du mode réseau défini dans la définition de tâche. Nous vous recommandons d'utiliser le mode réseau `awsvpc`, à moins que vous n'ayez besoin d'utiliser un mode réseau différent.

Les modes réseau disponibles sont les suivants.

Mode réseau	Conteneurs Linux sur EC2	Conteneurs Windows sur EC2	Description
`awsvpc`	Oui	Oui	La tâche reçoit sa propre interface réseau Elastic (ENI) et une adresse IPv4 ou IPv6 privée principale. Cela confère à la tâche les mêmes propriétés de réseaux que les instances Amazon EC2.
`bridge`	Oui	Non	La tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Linux utilise le pilote réseau Docker `bridge`. Ceci est le mode réseau par défaut sous Linux si aucun mode réseau n'est spécifié dans la définition de la tâche.
`host`	Oui	Non	La tâche utilise le réseau de l'hôte qui contourne le réseau virtuel intégré de Docker en mappant les ports de conteneur directement à l'ENI de l'instance Amazon EC2 qui héberge la tâche. Les mappages de ports dynamiques ne peuvent pas être utilisés dans ce mode réseau. Dans une définition de tâche qui utilise ce mode, un conteneur doit spécifier un numéro `hostPort` spécifique. Un numéro de port sur un hôte ne peut pas être utilisé par plusieurs tâches. Dans ce mode, vous ne pouvez pas exécuter plusieurs tâches de la même définition de tâche sur une instance Amazon EC2 unique.
`none`	Oui	Non	La tâche n'a pas de connectivité réseau externe.

Mode réseau	Conteneurs Linux sur EC2	Conteneurs Windows sur EC2	Description
`default`	Non	Oui	La tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Windows utilise le pilote réseau Docker `nat`. Ceci est le mode réseau par défaut sous Windows si aucun mode réseau n'est spécifié dans la définition de la tâche.

Pour plus d'informations sur la mise en réseau Docker sous Linux, consultez la section Présentation de la mise en réseau dans la Documentation Docker.

Pour plus d'informations sur la mise en réseau Docker sous Windows, consultez la section Mise en réseau des conteneurs Windows dans la Documentation relative aux conteneurs sous Windows de Microsoft.

Utilisation d'un VPC dans IPv6-only mode

Dans un IPv6-only configuration, vos tâches Amazon ECS communiquent exclusivement via IPv6. Pour configurer des VPC et des sous-réseaux pour un IPv6-only configuration, vous devez ajouter un bloc d'adresse CIDR IPv6 au VPC et créer de nouveaux sous-réseaux qui incluent uniquement un bloc d'adresse CIDR IPv6. Pour plus d'informations, consultez les sections Ajout du support IPv6 à votre VPC et Création d'un sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Vous devez également mettre à jour les tables de routage avec des cibles IPv6 et configurer les groupes de sécurité avec des règles IPv6. Pour plus d'informations, consultez les sections Configuration des tables de routage et Configuration des règles des groupes de sécurité dans le Guide de l'utilisateur Amazon VPC.

Les considérations suivantes s'appliquent :

- Vous pouvez mettre à jour un IPv4-only ou un service Amazon ECS dualstack vers un IPv6-only configuration en mettant à jour directement le service pour l'utiliser IPv6-only sous-réseaux ou en créant un parallèle IPv6-only service et en utilisant les déploiements bleu-vert d'Amazon ECS pour

transférer le trafic vers le nouveau service. Pour plus d'informations sur les déploiements bleu-vert Amazon ECS, consultez la section Amazon ECS blue/green déploiements.

- Un IPv6-only Le service Amazon ECS doit utiliser des équilibreurs de charge à double pile avec des groupes cibles IPv6. Si vous procédez à la migration d'un service Amazon ECS existant qui repose sur un Application Load Balancer ou un Network Load Balancer, vous pouvez créer un équilibreur de charge à double pile et lui transférer le trafic depuis l'ancien équilibreur de charge, ou mettre à jour le type d'adresse IP de l'équilibreur de charge existant.

  Pour plus d'informations sur les Network Load Balancer, consultez les sections Création d'un Network Load Balancer et Mise à jour des types d'adresses IP pour votre Network Load Balancer dans le Guide de l'utilisateur des Network Load Balancer. Pour plus d'informations sur les Application Load Balancer, consultez les sections Création d'un Application Load Balancer et Mise à jour des types d'adresses IP pour votre Application Load Balancer dans le Guide de l'utilisateur des Application Load Balancer.
- IPv6-only la configuration n'est pas prise en charge sur Windows. Vous devez utiliser Amazon ECS-optimized AMI Linux pour exécuter des tâches dans un IPv6-only configuration. Pour plus d'informations sur Amazon ECS-optimized AMI Linux, voir Amazon ECS-optimized AMI Linux.
- Lorsque vous lancez une instance de conteneur pour exécuter des tâches dans un IPv6-only configuration, vous devez définir une adresse IPv6 principale pour l'instance à l'aide du `--enable-primary-ipv6` Paramètre EC2.

  > **Note**
  >
  > Sans adresse IPv6 principale, les tâches exécutées sur l'instance de conteneur dans les modes réseau hôte ou pont échoueront à s'enregistrer auprès des équilibreurs de charge ou de AWS Cloud Map.

  Pour plus d'informations sur l'option `--enable-primary-ipv6` pour l'exécution d'instances Amazon EC2, consultez la section run-instances dans la Référence des commandes AWS CLI .

  Pour plus d'informations sur le lancement d'instances de conteneurs à l'aide du Console de gestion AWS, voir Lancement d'une instance de conteneur Amazon ECS Linux.
- Par défaut, l'agent de conteneur Amazon ECS essaie de détecter la compatibilité de l'instance de conteneur pour un IPv6-only configuration en examinant les routes IPv4 et IPv6 par défaut de l'instance. Pour annuler ce comportement, vous pouvez définir le paramètre

`ECS_INSTANCE_IP_COMPATIBILITY` sur `ipv4` ou `ipv6` dans le fichier `/etc/ecs/ecs.config` de l'instance.

- Les tâches doivent utiliser la version `1.99.1` ou une version ultérieure de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de l'agent utilisée par votre instance et la mise à jour si nécessaire, consultez la section Mise à jour de l'agent de conteneur Amazon ECS.
- Pour les tâches Amazon ECS dans un IPv6-only configuration avec laquelle communiquer IPv4-only points de terminaison, que vous pouvez configurer DNS64 et NAT64 pour la traduction d'adresses réseau d'IPv6 vers IPv4. Pour plus d'informations, veuillez consulter la section DNS64 et NAT64 dans le Guide de l'utilisateur Amazon VPC.
- Charges de travail Amazon ECS dans un IPv6-only la configuration doit utiliser les points de terminaison URI des images Amazon ECR dualstack lors de l'extraction d'images depuis Amazon ECR. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

> **Note**
>
> Amazon ECR ne prend pas en charge les points de terminaison VPC à interface dualstack qui effectuent des tâches dans un IPv6-only la configuration peut utiliser. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

- Amazon ECS Exec n'est pas pris en charge dans un IPv6-only configuration.

Régions AWS ce soutien IPv6-only mode pour Amazon ECS

Vous pouvez exécuter des tâches dans IPv6-only configuration dans la suivante AWS régions dans lesquelles Amazon ECS est disponible :

- US East (Ohio)
- USA Est (Virginie du Nord)
- 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)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)
- Canada Ouest (Calgary)
- Chine (Pékin)
- China (Ningxia)
- Europe (Francfort)
- Europe (Londres)
- Europe (Milan)
- Europe (Paris)
- Europe (Espagne)
- Israël (Tel Aviv)
- Middle East (Bahrain)
- Moyen-Orient (EAU)
- Amérique du Sud (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Allocation d’une interface réseau à une tâche Amazon ECS

Les fonctions de mise en réseau des tâches fournies par le mode réseau `awsvpc` attribuent aux tâches Amazon ECS les mêmes propriétés de mise en réseau que les instances Amazon EC2. L’utilisation du mode réseau `awsvpc` simplifie la mise en réseau des conteneurs, car vous disposez d’un meilleur contrôle sur la manière dont vos applications communiquent entre elles et avec d’autres services au sein de vos VPC. Le mode réseau `awsvpc` fournit également une sécurité accrue pour vos conteneurs en vous permettant d’utiliser des groupes de sécurité et des outils de surveillance

réseau à un niveau plus détaillé dans les tâches ECS. Vous pouvez également utiliser d'autres fonctionnalités réseau Amazon EC2, telles que les journaux de flux VPC, pour surveiller le trafic entrant et sortant de vos tâches. De plus, les conteneurs qui appartiennent à la même tâche peuvent communiquer via l'interface `localhost`.

L'interface réseau Elastic (ENI) est une fonctionnalité entièrement gérée d'Amazon ECS. Amazon ECS crée l'ENI et l'attache à l'instance Amazon EC2 hôte avec le groupe de sécurité spécifié. La tâche envoie et reçoit le trafic réseau sur l'ENI de la même manière que les instances Amazon EC2 avec leurs interfaces réseau principales. Chaque ENI de tâche se voit attribuer une adresse IPv4 privée par défaut. Si votre VPC est activé pour le mode double pile et que vous utilisez un sous-réseau avec un bloc CIDR IPv6, l'ENI de tâche recevra également une adresse IPv6. Chaque tâche ne peut avoir qu'une seule ENI.

Ces ENI sont visibles dans la console Amazon EC2 de votre compte. Les ENI ne peuvent être ni détachées ni modifiées depuis votre compte. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Vous pouvez consulter les informations relatives aux pièces jointes ENI pour les tâches dans la console Amazon ECS ou à l'aide du DescribeTasks Fonctionnement de l'API. Lorsque la tâche s'arrête ou si le service est réduit, l'ENI de la tâche est détachée et supprimée.

Lorsque vous avez besoin d'une densité ENI accrue, utilisez les paramètres du compte `awsvpcTrunking`. Amazon ECS crée et attache également une interface réseau « de jonction » pour votre instance de conteneur. Le réseau « de jonction » est entièrement géré par Amazon ECS. L'ENI « de jonction » est supprimée lorsque vous résiliez votre instance de conteneur ou lorsque vous annulez son enregistrement dans le cluster Amazon ECS. Pour plus d'informations sur le paramètre de compte `awsvpcTrunking`, consultez la section Conditions préalables.

Vous le spécifiez `awsvpc` dans le paramètre `networkMode` de la définition de tâche. Pour plus d'informations, consultez Mode réseau.

Ensuite, lorsque vous exécutez une tâche ou créez un service, utilisez le paramètre `networkConfiguration` qui comprend un ou plusieurs sous-réseaux dans lesquels placer vos tâches et un ou plusieurs groupes de sécurité à attacher à une ENI. Pour plus d'informations, consultez Configuration réseau. Les tâches sont placées sur des instances Amazon EC2 compatibles dans les mêmes zones de disponibilité que ces sous-réseaux et les groupes de sécurité spécifiés sont associés à l'ENI qui est allouée pour la tâche.

Considérations relatives à Linux

Tenez compte des éléments suivants lorsque vous utilisez le système d'exploitation Linux.

- Si vous utilisez une instance p5.48xlarge en mode `awsvpc`, vous ne pouvez pas exécuter plus d'une tâche sur l'instance.
- Tâches et services qui utilisent `awsvpc` le mode réseau nécessite le rôle lié au service Amazon ECS pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la Console de gestion AWS. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié au service en procédant comme suit AWS CLI commande :

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Votre instance Linux Amazon EC2 Linux nécessite une version `1.15.0` ou une version ultérieure de l'agent de conteneur pour exécuter des tâches qui utilisent le mode réseau `awsvpc`. Si vous utilisez un Amazon ECS-optimized AMI, votre instance a besoin d'au moins une version `1.15.0-4` de la `ecs-init` colis également.
- Amazon ECS renseigne le nom d'hôte de la tâche avec un Amazon-provided nom d'hôte DNS (interne) lorsque `enableDnsHostnames` et `enableDnsSupport` les options sont activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Les tâches Amazon ECS qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance Amazon EC2 qui l'héberge. Il existe un quota par défaut pour le nombre d'interfaces réseau qui peuvent être attachées à une instance Amazon EC2 Linux. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI pouvant lui être attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez attacher deux ENI supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Pour de plus amples informations sur les limites d'ENI par défaut pour chaque type d'instance, consultez la section Adresses IP privées par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.
- Amazon ECS prend en charge le lancement d'instances Amazon EC2 Linux avec une augmentation de la densité ENI et utilisant des types d'instances pris en charge. Lorsque vous activez le paramètre de compte `awsvpcTrunking` et enregistrez les instances Linux Amazon EC2 à l'aide de ces types d'instance dans votre cluster, ces instances ont des quotas ENI plus élevés. L'utilisation de ces instances avec ce quota plus élevé signifie que vous pouvez placer davantage

de tâches sur chaque instance Amazon EC2 Linux. Pour utiliser une densité ENI supérieure avec la fonctionnalité de jonction, vos instances Amazon EC2 doivent utiliser la version `1.28.1` ou ultérieure de l'agent de conteneur. Si vous utilisez un Amazon ECS-optimized AMI, votre instance nécessite également au moins une version `1.28.1-2` de la `ecs-init` colis. Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Accès aux fonctionnalités d'Amazon ECS avec les paramètres du compte. Pour en savoir plus sur la jonction ENI, consultez Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux.

- Lors de l'hébergement de tâches qui utilisent le mode réseau `awsvpc` sur les instances Linux Amazon EC2, les ENI de votre tâche ne reçoivent pas d'adresses IP publiques. Pour accéder à Internet, les tâches doivent être lancées dans un sous-réseau privé configuré pour utiliser une passerelle NAT. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC. L'accès entrant au réseau doit se faire depuis un VPC avec l'adresse IP privée ou doit transiter par un équilibreur de charge au sein du VPC. Les tâches lancées dans les sous-réseaux publics n'ont pas accès à Internet.
- Amazon ECS reconnaît uniquement les ENI qu'il attache à vos instances Amazon EC2 Linux. Si vous avez associé manuellement des ENI à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment de cartes réseau. Cela peut entraîner l'expiration de la tâche et le passage à un état de déprovisionnement, puis à un état d'arrêt. Nous vous recommandons de ne pas attacher manuellement des ENI à vos instances.
- Les instances Amazon EC2 Linux doivent être enregistrées avec la capacité `ecs.capability.task-eni` pour pouvoir faire l'objet d'un placement des tâches avec le mode réseau `awsvpc`. Les instances exécutant la version `1.15.0-4` ou une version ultérieure d'`ecs-init` sont automatiquement enregistrées avec cet attribut.
- Les ENI créées et attachées à vos instances Amazon EC2 Linux ne peuvent pas être détachées manuellement ni modifiées par votre compte. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- Il existe une limite de 16 sous-réseaux et 5 groupes de sécurité pouvant être spécifiés dans `awsVpcConfiguration` lors de l'exécution d'une tâche ou de la création d'un service qui utilise le mode réseau `awsvpc`. Pour plus d'informations, voir AwsVpcConfiguration dans le Référence de l'API Amazon Elastic Container Service.
- Lorsqu'une tâche est démarrée avec le mode réseau `awsvpc`, l'agent de conteneur Amazon ECS crée un conteneur `pause` supplémentaire pour chaque tâche avant de démarrer les conteneurs dans la définition de tâche. Ensuite, il configure l'espace de noms réseau du conteneur `pause` en exécutant les plugins CNI amazon-ecs-cni-plugins. L'agent démarre ensuite le reste des

conteneurs dans la tâche afin qu'ils partagent la pile réseau du conteneur `pause`. Cela signifie que tous les conteneurs d'une tâche peuvent être adressés par les adresses IP de l'ENI et qu'ils peuvent communiquer entre eux via l'interface `localhost`.

- Les services avec des tâches qui utilisent le mode réseau `awsvpc` prennent uniquement en charge Application Load Balancer et Network Load Balancer. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir `ip` comme type de cible. N'utilisez pas `instance`. En effet, les tâches qui utilisent le mode réseau `awsvpc` sont associées à une ENI, et non à une instance Amazon EC2 Linux. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.
- Si votre VPC est mis à jour pour modifier le jeu d'options DHCP qu'il utilise, vous ne pouvez pas appliquer ces modifications aux tâches existantes. Commencez de nouvelles tâches en leur appliquant ces modifications, vérifiez qu'elles fonctionnent correctement, puis arrêtez les tâches existantes afin de modifier ces configurations réseau en toute sécurité.

Considérations relatives à Windows

Voici quelques points à prendre en compte lorsque vous utilisez le système d'exploitation Windows :

- Les instances de conteneur utilisant l'AMI Windows Server 2016 optimisée pour Amazon ECS ne peuvent pas héberger des tâches qui utilisent le mode réseau `awsvpc`. Si vous disposez d'un cluster qui contient des AMI Windows Server 2016 optimisées pour Amazon ECS et des AMI Windows qui prennent en charge le mode réseau `awsvpc`, les tâches qui utilisent le mode réseau `awsvpc` ne sont pas lancées sur les instances Windows 2016 Server. Elles sont plutôt lancées sur des instances qui prennent en charge le mode réseau `awsvpc`.
- Votre instance Windows Amazon EC2 nécessite une version `1.57.1` ou une version ultérieure de l'agent contenant à utiliser CloudWatch métriques pour les conteneurs Windows qui utilisent le `awsvpc` mode réseau.
- Tâches et services qui utilisent `awsvpc` le mode réseau nécessite le rôle lié au service Amazon ECS pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans Console de gestion AWS. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié au service en procédant comme suit AWS CLI commande.

```
aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
```

- Votre instance Amazon EC2 Windows nécessite une version `1.54.0` ou une version ultérieure de l'agent de conteneur pour exécuter des tâches qui utilisent le mode réseau `awsvpc`. Lorsque vous amorcez l'instance, vous devez configurer les options requises pour le mode réseau `awsvpc`. Pour plus d'informations, consultez the section called “Amorçage des instances de conteneur”.
- Amazon ECS remplit le nom d'hôte d'une tâche avec un nom d'hôte DNS (interne) fourni par Amazon lorsque les options `enableDnsHostnames` et `enableDnsSupport` sont toutes deux activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Les tâches Amazon ECS qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance Amazon EC2 Windows qui l'héberge. Il existe un quota par défaut pour le nombre d'interfaces réseau pouvant être attachées à une instance Amazon EC2 Windows. L'interface réseau principale est considérée comme une seule pour ce quota. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI qui lui sont attachées. L'interface réseau principale de l'instance compte parmi celles-ci. Vous pouvez attacher deux ENI supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Pour de plus amples informations sur les limites d’ENI par défaut pour chaque type d’instance, consultez la section Adresses IP privées par interface réseau et par type d’instance dans le Guide de l’utilisateur Amazon EC2.
- Lors de l'hébergement de tâches qui utilisent le mode réseau `awsvpc` sur les instances Windows Amazon EC2, les ENI de votre tâche ne reçoivent pas d'adresses IP publiques. Pour accéder à Internet, lancez les tâches dans un sous-réseau privé configuré pour utiliser une passerelle NAT. Pour plus d’informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l’utilisateur Amazon VPC. L'accès entrant au réseau doit se faire depuis le VPC avec l'adresse IP privée ou doit transiter par un équilibreur de charge au sein du VPC. Les tâches lancées dans les sous-réseaux publics n'ont pas accès à Internet.
- Amazon ECS reconnaît uniquement les ENI qu'il a attachées à votre instance Windows Amazon EC2. Si vous avez associé manuellement des ENI à vos instances, Amazon ECS peut tenter d'ajouter une tâche à une instance qui ne dispose pas de suffisamment de cartes réseau. Cela peut entraîner l'expiration de la tâche et le passage à un état de déprovisionnement, puis à un état d'arrêt. Nous vous recommandons de ne pas attacher manuellement des ENI à vos instances.
- Les instances Amazon EC2 Windows doivent être enregistrées avec l'interface réseau `ecs.capability.task-eni` pour pouvoir faire l'objet d'un placement des tâches avec le mode réseau `awsvpc`.

- Vous ne pouvez pas modifier ou détacher manuellement les ENI créées et attachées à vos instances Windows Amazon EC2. Cela vous évite de supprimer accidentellement une ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- Vous pouvez uniquement spécifier jusqu'à 16 sous-réseaux et 5 groupes de sécurité dans `awsVpcConfiguration`, lorsque vous exécutez une tâche ou créez un service qui utilise le mode réseau `awsvpc`. Pour plus d'informations, voir AwsVpcConfiguration dans le Référence de l'API Amazon Elastic Container Service.
- Lorsqu'une tâche est démarrée avec le mode réseau `awsvpc`, l'agent de conteneur Amazon ECS crée un conteneur `pause` supplémentaire pour chaque tâche avant de démarrer les conteneurs dans la définition de tâche. Ensuite, il configure l'espace de noms réseau du conteneur `pause` en exécutant les plugins CNI amazon-ecs-cni-plugins. L'agent démarre ensuite le reste des conteneurs dans la tâche afin qu'ils partagent la pile réseau du conteneur `pause`. Cela signifie que tous les conteneurs d'une tâche peuvent être adressés par les adresses IP de l'ENI et qu'ils peuvent communiquer entre eux via l'interface `localhost`.
- Les services avec des tâches qui utilisent le mode réseau `awsvpc` prennent uniquement en charge Application Load Balancer et Network Load Balancer. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. En effet, les tâches qui utilisent le mode réseau `awsvpc` sont associées à une ENI, et non à une instance Amazon EC2 Windows. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.
- Si votre VPC est mis à jour pour modifier le jeu d'options DHCP qu'il utilise, vous ne pouvez pas appliquer ces modifications aux tâches existantes. Commencez de nouvelles tâches en leur appliquant ces modifications, vérifiez qu'elles fonctionnent correctement, puis arrêtez les tâches existantes afin de modifier ces configurations réseau en toute sécurité.
- Les éléments suivants ne sont pas pris en charge lorsque vous utilisez le mode réseau `awsvpc` dans une configuration Windows EC2 :
  - Dual-stack configuration
  - IPv6
  - Jonction ENI

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer via IPv4, IPv6 ou les deux. Les adresses IPv4 et IPv6 sont indépendantes les unes des autres. Vous devez donc configurer le routage et la sécurité dans votre VPC séparément pour IPv4 et IPv6. Pour plus

d'informations sur la configuration de votre VPC pour le mode double pile, consultez la rubrique Migration vers IPv6 dans le Guide de l'utilisateur Amazon VPC.

Si vous avez configuré votre VPC avec une passerelle Internet ou une passerelle Internet sortante uniquement, vous pouvez utiliser votre VPC en mode double pile. De ce fait, les tâches auxquelles une adresse IPv6 est attribuée peuvent accéder à Internet via une passerelle Internet ou une passerelle Internet de sortie uniquement. Les passerelles NAT sont facultatives. Pour plus d'informations, voir Passerelles Internet et Egress-only passerelles Internet dans le Guide de l'utilisateur d'Amazon VPC.

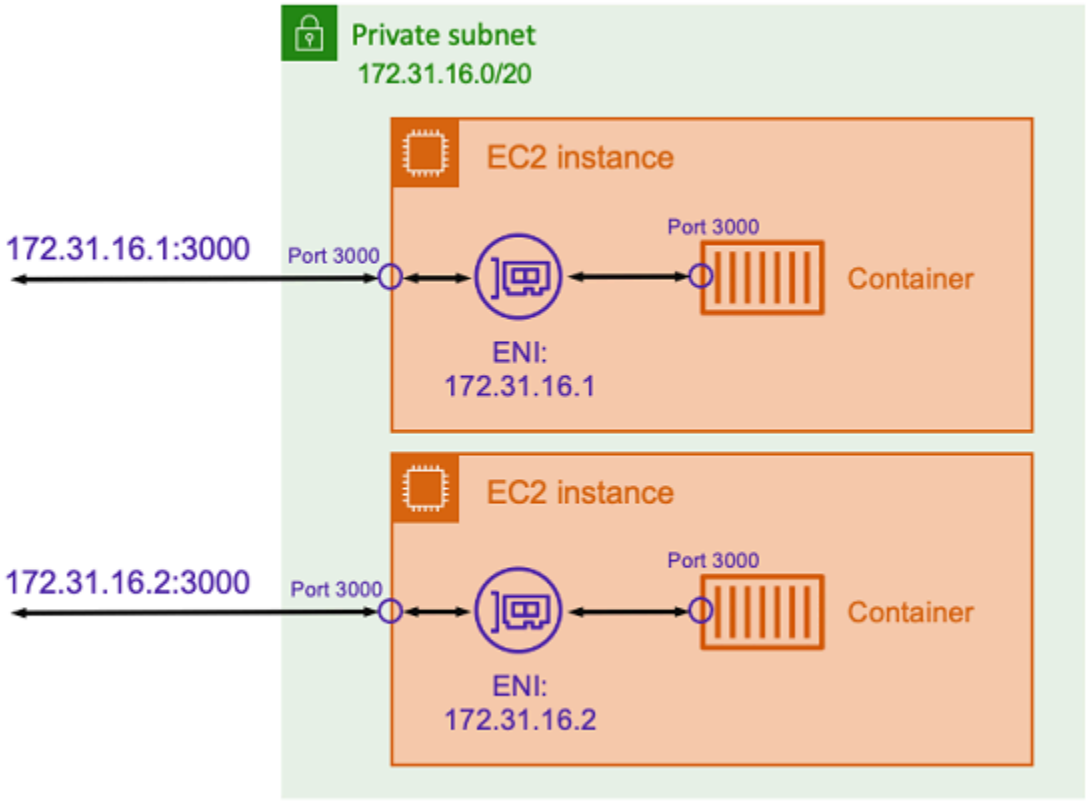
Une adresse IPv6 est attribuée aux tâches Amazon ECS si les conditions suivantes sont réunies :

- L'instance Linux Amazon EC2 hébergeant la tâche utilise la version `1.45.0` ou version ultérieure de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de l'agent utilisée par votre instance et la mise à jour si nécessaire, veuillez consulter Mise à jour de l'agent de conteneur Amazon ECS.
- Le paramètre de compte `dualStackIPv6` est activé. Pour plus d'informations, consultez Accès aux fonctionnalités d'Amazon ECS avec les paramètres du compte.
- Votre tâche utilise le mode réseau `awsvpc`.
- Votre VPC et votre sous-réseau sont configurés pour IPv6. La configuration inclut les interfaces réseau créées dans le sous-réseau spécifié. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, veuillez consulter les rubriques Migration vers IPv6 et Modifier l'attribut d'adressage IPv6 de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Mappage des ports de conteneurs Amazon ECS vers l'interface réseau de l'instance EC2

Le mode réseau `host` est uniquement pris en charge pour les tâches Amazon ECS hébergées sur des instances Amazon EC2. Il n'est pas pris en charge lors de l'utilisation d'Amazon ECS sur Fargate.

Le mode réseau `host` est le mode réseau le plus basique pris en charge dans Amazon ECS. En mode hôte, la mise en réseau du conteneur est liée directement à l'hôte sous-jacent qui exécute le conteneur.



Supposons que vous exécutez un Node.js conteneur avec une application Express qui écoute sur le port `3000` similaire à celui illustré dans le schéma précédent. Lorsque le mode réseau `host` est utilisé, le conteneur reçoit le trafic sur le port 3 000 en utilisant l'adresse IP de l'instance Amazon EC2 de l'hôte sous-jacent. Nous vous recommandons de ne pas utiliser ce mode.

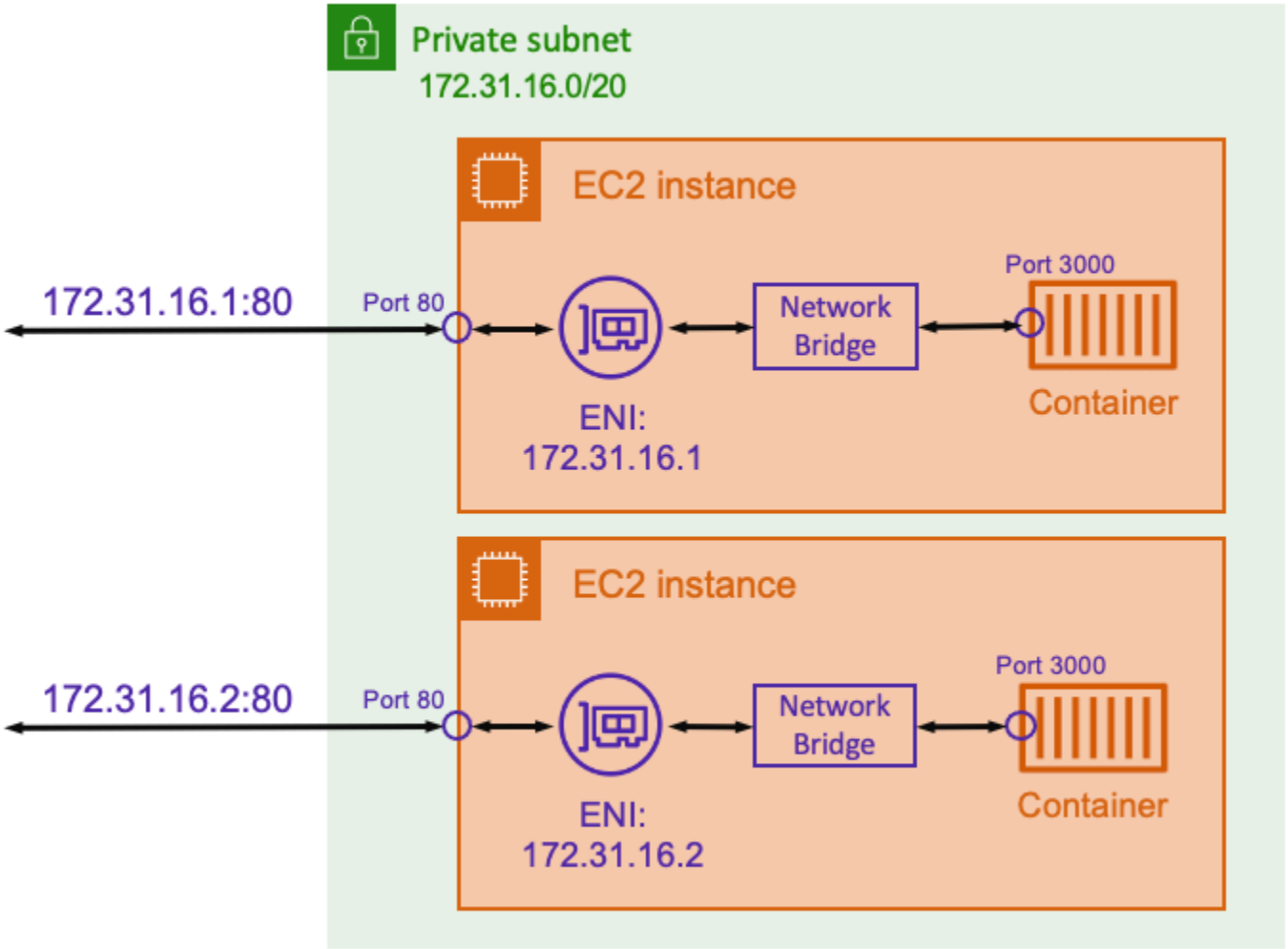
L'utilisation de ce mode réseau présente des inconvénients importants. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait que seule la première tâche peut se lier au port requis sur l'instance Amazon EC2. Il est également impossible de remapper un port de conteneur lorsqu'il utilise le mode réseau `host`. Par exemple, si une application doit écouter un numéro de port spécifique, vous ne pouvez pas remapper le numéro de port directement. Vous devez plutôt gérer les conflits de ports en modifiant la configuration de l'application.

L'utilisation du mode réseau `host` a également des implications en termes de sécurité. Ce mode permet aux conteneurs de se faire passer pour l'hôte et de se connecter à des services réseau privés en boucle sur l'hôte.

Utilisation Dockerréseau virtuel pour les tâches Amazon ECS Linux

Le mode réseau `bridge` est uniquement pris en charge pour les tâches Amazon ECS hébergées sur des instances Amazon EC2.

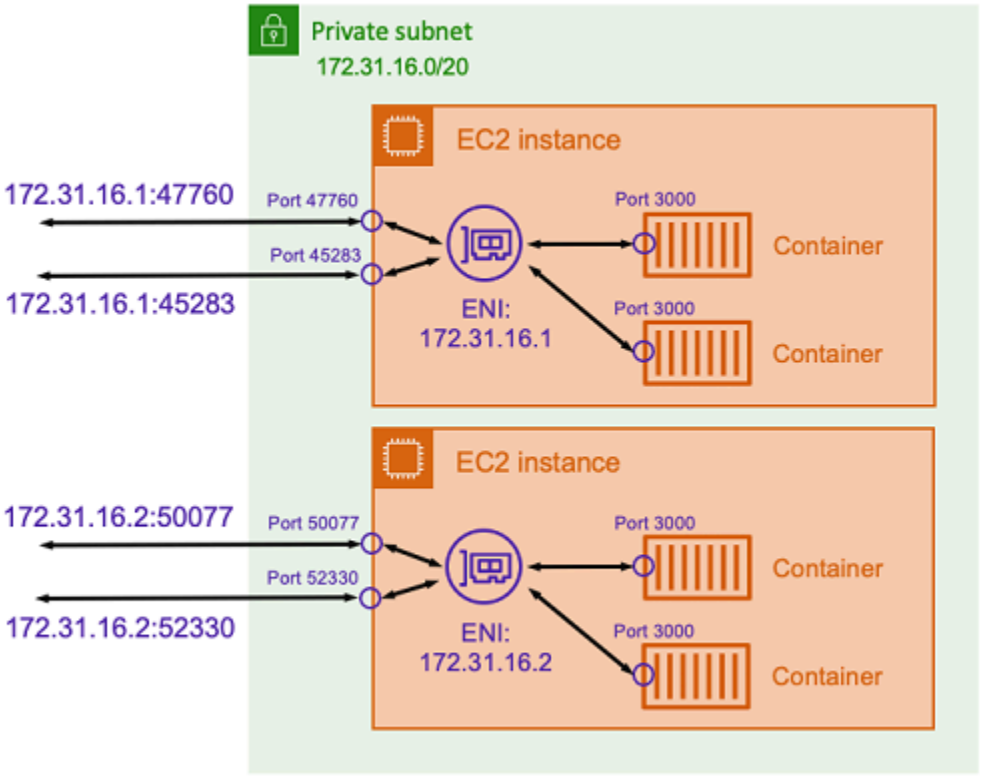
Avec le mode `bridge`, vous utilisez un pont réseau virtuel pour créer une couche entre l'hôte et la mise en réseau du conteneur. De cette façon, vous pouvez créer des mappages de ports qui remappent un port hôte à un port de conteneur. Les mappages peuvent être statiques ou dynamiques.



Avec un mappage statique des ports, vous pouvez définir explicitement le port hôte que vous souhaitez mapper à un port de conteneur. En utilisant l'exemple précédent, port `80` sur l'hôte est mappé au port `3000` sur le contenant. Pour communiquer avec l'application conteneurisée, vous envoyez le trafic au port `80` vers l'adresse IP de l'instance Amazon EC2. Du point de vue de l'application conteneurisée, le trafic entrant est visible sur le port `3000`.

Si vous souhaitez uniquement modifier le port de trafic, les mappages statiques de ports conviennent. Cependant, cela présente toujours le même inconvénient que l'utilisation du mode réseau `host`. Vous ne pouvez exécuter qu'une seule instanciation d'une tâche sur chaque hôte. Cela est dû au fait qu'un mappage statique de port ne permet de mapper qu'un seul conteneur sur le port 80.

Pour résoudre ce problème, pensez à utiliser le mode réseau `bridge` avec un mappage dynamique des ports, comme indiqué dans le schéma suivant.



En ne spécifiant pas de port hôte dans le mappage des ports, vous pouvez demander à Docker de choisir un port aléatoire inutilisé dans la plage de ports éphémères et de l'attribuer comme port hôte public pour le conteneur. Par exemple, le Node.js écoute de l'application sur le port `3000` sur le conteneur peut être attribué un port aléatoire à numéro élevé tel que `47760` sur l'hôte Amazon EC2. Cela signifie que vous pouvez exécuter plusieurs copies de ce conteneur sur l'hôte. De plus, chaque conteneur peut se voir attribuer son propre port sur l'hôte. Chaque copie du conteneur reçoit du trafic au port `3000`. Toutefois, les clients qui envoient du trafic vers ces conteneurs utilisent les ports hôtes assignés de manière aléatoire.

Amazon ECS vous aide à suivre les ports assignés de manière aléatoire pour chaque tâche. Pour ce faire, il met automatiquement à jour les groupes cibles de l'équilibreur de charge et AWS Cloud

Map découverte de services pour avoir la liste des adresses IP et des ports des tâches. Cela facilite l'utilisation des services fonctionnant en mode `bridge` utilisant des ports dynamiques.

Cependant, l'un des inconvénients de l'utilisation du mode réseau `bridge` est qu'il est difficile de verrouiller les communications de service à service. Comme les services peuvent être affectés à n'importe quel port aléatoire et inutilisé, il est nécessaire d'ouvrir de larges plages de ports entre les hôtes. Cependant, il n'est pas facile de créer des règles spécifiques afin qu'un service particulier ne puisse communiquer qu'avec un autre service spécifique. Les services ne disposent d'aucun port spécifique à utiliser pour les règles de mise en réseau des groupes de sécurité.

Configuration du mode réseau bridge pour IPv6-only charges de travail

Pour configurer le mode `bridge` pour la communication via IPv6, vous devez mettre à jour les paramètres du démon Docker. Mettez à jour `/etc/docker/daemon.json` les éléments suivants :

```
{
  "ipv6": true,
  "fixed-cidr-v6": "2001:db8:1::/64",
  "ip6tables": true,
  "experimental": true
}
```

Après avoir mis à jour les paramètres du démon Docker, vous devez le redémarrer.

> **Note**
>
> Lorsque vous mettez à jour et redémarrez le démon Docker, le transfert IPv6 est activé sur l'instance, ce qui peut entraîner la perte des routes par défaut sur les instances qui utilisent une AMI Amazon Linux 2. Pour éviter cela, utilisez la commande suivante pour ajouter une route par défaut via la passerelle IPv6 du sous-réseau.
>
> ```
> ip route add default via FE80:EC2::1 dev eth0 metric 100
> ```

Options de mise en réseau des tâches Amazon ECS pour Fargate

Par défaut, chaque tâche Amazon ECS sur Fargate reçoit une interface réseau Elastic (ENI) avec une adresse IP privée principale. Lorsque vous utilisez un sous-réseau public, vous pouvez

éventuellement attribuer une adresse IP publique à l'ENI de la tâche. Si votre VPC est configuré pour le mode double pile et que vous utilisez un sous-réseau avec un bloc CIDR IPv6, l'ENI de votre tâche reçoit également une adresse IPv6. Une tâche ne peut avoir qu'une seule ENI associée à la fois. Les conteneurs qui appartiennent à la même tâche peuvent également communiquer via l'interface `localhost`. Pour plus d'informations sur les VPC et les sous-réseaux, consultez la section Fonctionnement d'Amazon VPC dans le Guide de l'utilisateur Amazon VPC.

Pour qu'une tâche sur Fargate puisse extraire une image de conteneur, la tâche doit avoir un routage vers Internet. L'exemple suivant décrit également comment vous pouvez vérifier le routage vers Internet pour votre tâche.

- Lorsque vous utilisez un sous-réseau public, vous pouvez attribuer une adresse IP publique à l'ENI de la tâche.
- Lorsque vous utilisez un sous-réseau privé, une passerelle NAT peut être attachée au sous-réseau.
- Lorsque vous utilisez des images de conteneur hébergées dans Amazon ECR, vous pouvez configurer Amazon ECR pour utiliser un point de terminaison VPC d'interface et l'extraction de l'image se produit sur l'adresse IPv4 privée de la tâche. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Dans la mesure où chaque tâche obtient sa propre ENI, vous pouvez utiliser des fonctionnalités de mise en réseau telles que les journaux de flux VPC, que vous pouvez utiliser pour surveiller le trafic vers et depuis vos tâches. Pour plus d'informations, consultez Journaux de flux VPC dans le Guide de l'utilisateur Amazon VPC.

Vous pouvez également profiter de AWS PrivateLink. Vous pouvez configurer un point de terminaison d'interface VPC afin de pouvoir accéder aux API Amazon ECS via des adresses IP privées. AWS PrivateLink limite tout le trafic réseau entre votre VPC et Amazon ECS vers le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle. Pour plus d'informations, consultez la section Points de terminaison de VPC d'interface Amazon ECS (AWS PrivateLink).

Pour des exemples d'utilisation du `NetworkConfiguration` ressource avec CloudFormation, voir the section called "Exemples de CloudFormation modèles".

Les ENI créées sont entièrement gérées par AWS Fargate. De plus, une politique IAM associée est utilisée pour accorder des autorisations pour Fargate. Pour les tâches utilisant la version `1.4.0` de

la plateforme Fargate ou une version ultérieure, la tâche reçoit une seule ENI (appelée ENI de tâche) et tout le trafic réseau passe par cette ENI au sein de votre VPC. Ce trafic est enregistré dans les journaux de flux de votre VPC. Pour les tâches qui utilisent la version `1.3.0` de la plateforme Fargate et les versions antérieures, en plus de l'ENI de la tâche, la tâche reçoit également une ENI distincte détenue par Fargate, qui est utilisée pour certains trafics réseau qui ne sont pas visibles dans les journaux de flux VPC. Le tableau suivant décrit le comportement du trafic réseau et la stratégie IAM requise pour chaque version de plateforme.

Action	Flux de trafic avec les versions **`1.3.0`** et antérieures de la plateforme Linux	Flux de trafic avec la version **`1.4.0`** de la plateforme Linux	Flux de trafic avec la version **`1.0.0`** de la plateforme Windows	Autorisation IAM
Récupération des informations d'identification de connexion Amazon ECR	ENI Fargate	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Extraction d'image	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Envoi de journaux via un pilote de journal	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche
Envoi de journaux via FireLens pour Amazon ECS	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM de tâche
Récupération de secrets à partir de Secrets Manager ou	ENI Fargate	ENI de tâche	ENI de tâche	Rôle IAM d'exécution de tâche

Action	Flux de trafic avec les versions **1.3.0** et antérieures de la plateforme Linux	Flux de trafic avec la version **1.4.0** de la plateforme Linux	Flux de trafic avec la version **1.0.0** de la plateforme Windows	Autorisation IAM
de Systems Manager				
Trafic du système de fichiers Amazon EFS	Non disponible	ENI de tâche	ENI de tâche	Rôle IAM de tâche
Trafic d'application	ENI de tâche	ENI de tâche	ENI de tâche	Rôle IAM de tâche

Considérations

Tenez compte des éléments suivants lorsque vous utilisez le réseau de tâches.

- Le rôle lié au service Amazon ECS est requis pour fournir à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la Console de gestion AWS. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié au service à l'aide des méthodes suivantes AWS CLI commande.

  ```
  aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
  ```

- Amazon ECS remplit le nom d'hôte de la tâche avec un nom d'hôte DNS fourni par Amazon lorsque les options `enableDnsHostnames` et `enableDnsSupport` sont activées sur votre VPC. Si ces options ne sont pas activées, le nom d'hôte DNS de la tâche est un nom d'hôte aléatoire. Pour plus d'informations sur les paramètres DNS d'un VPC, veuillez consulter la rubrique Utilisation de DNS avec votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Vous pouvez uniquement spécifier jusqu'à 16 sous-réseaux et 5 groupes de sécurité pour `awsVpcConfiguration`. Pour plus d'informations, voir AwsVpcConfiguration dans le Référence de l'API Amazon Elastic Container Service.

- Vous ne pouvez pas détacher ou modifier manuellement les ENI créées et attachées par Fargate. L'objectif est d'empêcher la suppression accidentelle de toute ENI associée à une tâche en cours d'exécution. Pour libérer les ENI d'une tâche, arrêtez cette dernière.
- Si un sous-réseau VPC est mis à jour pour modifier l'ensemble d'options DHCP qu'il utilise, vous ne pouvez pas également appliquer ces modifications aux tâches existantes qui utilisent le VPC. Démarrez de nouvelles tâches, qui recevront le nouveau paramètre pour migrer en douceur tout en testant la nouvelle modification, puis arrêtez les anciennes, si aucune restauration n'est requise.
- Ce qui suit s’applique aux tâches exécutées sur la version de plateforme Fargate `1.4.0` ou ultérieure pour Linux ou `1.0.0` pour Windows. Les tâches lancées dans des sous-réseaux à double pile reçoivent une adresse IPv4 et une adresse IPv6. Tâches lancées dans IPv6-only les sous-réseaux reçoivent uniquement une adresse IPv6.
- Pour les tâches utilisant la version `1.4.0` de la plateforme ou ultérieure pour Linux ou `1.0.0` pour Windows, les ENI de tâche prennent en charge les trames jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficacité. La prise en charge des trames jumbo limite la surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo.
- Les services comportant des tâches qui utilisent Fargate ne prennent en charge que les équilibreurs de charge Application Load Balancer et Network Load Balancer. L'équilibreur de charge Classic Load Balancer n'est pas pris en charge. Lorsque vous créez des groupes cibles pour ces services, vous devez choisir le type de cible `ip`, et non `instance`. Pour plus d’informations, consultez Utilisation de l’équilibrage de charge pour répartir le trafic des services Amazon ECS.

Utilisation d'un VPC en mode double pile

Lorsque vous utilisez un VPC en mode double pile, vos tâches peuvent communiquer via IPv4, IPv6 ou les deux. Les adresses IPv4 et IPv6 sont indépendantes les unes des autres et vous devez configurer le routage et la sécurité dans votre VPC séparément pour IPv4 et IPv6. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, veuillez consulter la rubrique Migration vers IPv6 dans le Guide de l'utilisateur Amazon VPC.

Si les conditions suivantes sont réunies, des tâches Amazon ECS sur Fargate sont attribuées à une adresse IPv6 :

- Le paramètre de votre compte Amazon ECS `dualStackIPv6` est activé (`enabled`) pour que le principal IAM lance vos tâches dans la région dans laquelle vous les lancez. Ce paramètre ne peut être modifié qu'à l'aide de l'API ou AWS CLI. Vous avez la possibilité d'activer ce paramètre pour un principal IAM spécifique sur votre compte ou pour l'ensemble de votre compte en définissant les paramètres par défaut de votre compte. Pour plus d'informations, consultez Accès aux fonctionnalités d'Amazon ECS avec les paramètres du compte.
- Votre VPC et votre sous-réseau sont activés pour IPv6. Pour plus d'informations sur la configuration de votre VPC pour le mode double pile, consultez la rubrique Migration vers IPv6 dans le Guide de l'utilisateur Amazon VPC.
- Votre sous-réseau est activé pour l'attribution automatique d'adresses IPV6. Pour plus d'informations sur la configuration de votre sous-réseau, veuillez consulter Modifier l'attribut d'adressage IPv6 de votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.
- La tâche ou le service utilise la version de plateforme `1.4.0` Fargate ou une version ultérieure pour Linux.

Pour les tâches Amazon ECS exécutées sur Fargate dans un VPC en mode double pile, afin de communiquer avec les services de dépendance utilisés dans le processus de lancement des tâches tels que ECR, SSM et SecretManager, la table de routage du sous-réseau public nécessite IPv4 (0.0.0.0/0) route vers une passerelle Internet et la table de routage du sous-réseau privé nécessite IPv4 (0.0.0).0/0) route vers une passerelle NAT. Pour plus d'informations, voir Passerelles Internet et passerelles NAT dans le Guide de l'utilisateur d'Amazon VPC.

Pour obtenir des exemples de configuration d'un VPC à double pile, consultez Exemple de configuration VPC à double pile.

Utilisation d'un VPC dans IPv6-only mode

Dans un IPv6-only configuration, vos tâches Amazon ECS communiquent exclusivement via IPv6. Pour configurer des VPC et des sous-réseaux pour un IPv6-only configuration, vous devez ajouter un bloc d'adresse CIDR IPv6 au VPC et créer des sous-réseaux qui incluent uniquement un bloc d'adresse CIDR IPv6. Pour plus d'informations, consultez les sections Ajout du support IPv6 à votre VPC et Création d'un sous-réseau dans le Guide de l'utilisateur Amazon VPC. Vous devez également mettre à jour les tables de routage avec des cibles IPv6 et configurer les groupes de sécurité avec des règles IPv6. Pour plus d'informations, consultez les sections Configuration des tables de routage et Configuration des règles des groupes de sécurité dans le Guide de l'utilisateur Amazon VPC.

Les considérations suivantes s'appliquent :

- Vous pouvez mettre à jour un IPv4-only ou un service Amazon ECS dualstack vers un IPv6-only configuration en mettant à jour directement le service pour l'utiliser IPv6-only sous-réseaux ou en créant un parallèle IPv6-only service et en utilisant les déploiements bleu-vert d'Amazon ECS pour transférer le trafic vers le nouveau service. Pour plus d'informations sur les déploiements bleu-vert Amazon ECS, consultez la section Amazon ECS blue/green déploiements.
- Un IPv6-only Le service Amazon ECS doit utiliser des équilibreurs de charge à double pile avec des groupes cibles IPv6. Si vous procédez à la migration d'un service Amazon ECS existant qui repose sur un Application Load Balancer ou un Network Load Balancer, vous pouvez créer un équilibreur de charge à double pile et lui transférer le trafic depuis l'ancien équilibreur de charge, ou mettre à jour le type d'adresse IP de l'équilibreur de charge existant.

  Pour plus d'informations sur les Network Load Balancer, consultez les sections Création d'un Network Load Balancer et Mise à jour des types d'adresses IP pour votre Network Load Balancer dans le Guide de l'utilisateur des Network Load Balancer. Pour plus d'informations sur les Application Load Balancer, consultez les sections Création d'un Application Load Balancer et Mise à jour des types d'adresses IP pour votre Application Load Balancer dans le Guide de l'utilisateur des Application Load Balancer.
- IPv6-only la configuration n'est pas prise en charge sur Windows.
- Pour les tâches Amazon ECS dans un IPv6-only configuration avec laquelle communiquer IPv4-only points de terminaison, que vous pouvez configurer DNS64 et NAT64 pour la traduction d'adresses réseau d'IPv6 vers IPv4. Pour plus d'informations, veuillez consulter la section DNS64 et NAT64 dans le Guide de l'utilisateur Amazon VPC.
- IPv6-only la configuration est prise en charge sur la version de la plateforme Fargate `1.4.0` ou plus tard.
- Charges de travail Amazon ECS dans un IPv6-only la configuration doit utiliser les points de terminaison URI des images Amazon ECR dualstack lors de l'extraction d'images depuis Amazon ECR. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

> **Note**
>
> Amazon ECR ne prend pas en charge les points de terminaison VPC à interface dualstack qui effectuent des tâches dans un IPv6-only la configuration peut utiliser. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

- Amazon ECS Exec n'est pas pris en charge dans un IPv6-only configuration.
- Amazon CloudWatch ne prend pas en charge un point de terminaison FIPS à double pile qui peut être utilisé pour surveiller les tâches Amazon ECS dans IPv6-only configuration qui utilise FIPS-140 conformité. Pour plus d'informations sur FIPS-140, voir AWS Fargate Norme fédérale de traitement de l'information (FIPS-140).

Régions AWS ce soutien IPv6-only mode pour Amazon ECS

Vous pouvez exécuter des tâches dans IPv6-only configuration dans la suivante Régions AWS qu'Amazon ECS est disponible dans :

- US East (Ohio)
- USA Est (Virginie du Nord)
- 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)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Tokyo)
- Canada (Centre)
- Canada Ouest (Calgary)
- Chine (Pékin)
- China (Ningxia)
- Europe (Francfort)
- Europe (Londres)
- Europe (Milan)

- Europe (Paris)
- Europe (Espagne)
- Israël (Tel Aviv)
- Middle East (Bahrain)
- Moyen-Orient (EAU)
- Amérique du Sud (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Options de stockage pour les tâches Amazon ECS

Amazon ECS vous offre des options de stockage de données flexibles, économiques et faciles à utiliser, en fonction de vos besoins. Amazon ECS prend en charge les options de volume de données suivantes pour les conteneurs :

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon Elastic Block Store (Amazon EBS)	Fargate, Amazon EC2, instances gérées Amazon ECS	Linux, Windows (sur Amazon EC2 uniquement)	Peut être conservé lorsqu'il est associé à une tâche autonome. Éphémère lorsqu'il est rattaché à une tâche gérée par un service.	Les volumes Amazon EBS offrent un stockage par blocs économique, durable et hautement performant pour les charges de travail conteneurisées gourmandes en données. Les cas d'utilisation courants sont notamment les charges de

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				travail transacti onnelles telles que les bases de données, les bureaux virtuels et les volumes racine, ainsi que les charges de travail à débit intensif telles que le traitemen t des journaux et les charges de travail ETL. Pour plus d'informa tions, consultez Utilisation des volumes Amazon EBS avec Amazon ECS.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon Elastic File System (Amazon EFS)	Fargate, Amazon EC2, instances gérées Amazon ECS	Linux	Persistante	Les volumes Amazon EFS fournissent un stockage de fichiers partagé simple, doté d'une capacité de mise à l'échelle et persistant pour une utilisation avec vos tâches Amazon ECS. Ils croissent et diminuent automatiq uement à mesure que vous ajoutez ou supprimez des fichiers. Les volumes Amazon EFS prennent en charge la simultanéité et trouvent leur utilité dans les applications conteneurisées qui procèdent à une mise à l'échelle

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				horizontale et nécessitent le recours à des fonctionnalités de stockage telles qu'une faible latence, un débit élevé et une cohérence en lecture après écriture. Les cas d'utilisation courants sont notamment les charges de travail telles que l'analytique des données, le traitement multimédia, la gestion de contenu et les services Web. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon FSx pour Windows File Server	Amazon EC2	Windows	Persistante	Les volumes FSx for Windows File Server fournissent des serveurs de fichiers Windows entièrement gérés que vous pouvez utiliser pour l'allocation de vos tâches Windows nécessitant un stockage de fichiers persistant, distribué, partagé et statique. Les cas d'utilisation courants sont notamment les applications .NET qui peuvent nécessiter des dossiers locaux comme stockage persistant pour enregistrer les résultats des applications. Amazon FSx for

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				Windows File Server propose un dossier local dans le conteneur qui permet à plusieurs conteneurs d'effectuer des opérations de lecture-écriture sur le même système de fichiers pris en charge par un partage SMB. Pour plus d'informations, consultez Utilisation de FSx for Windows File Server avec Amazon ECS.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Amazon FSx pour NetApp AU ROBINET	Amazon EC2 (montage natif) ; Fargate (via des points d'accès S3 pour FSx)	Linux	Persistante	Amazon FSx pour NetApp Les volumes ONTAP fournissent une gestion complète NetApp Systèmes de fichiers ONTAP que vous pouvez utiliser pour provisionner vos tâches Linux nécessitant un stockage de fichiers partagé permanent , performan t et riche en fonctionnalités. Amazon FSx pour NetApp ONTAP prend en charge les protocoles NFS et SMB et fournit des fonctionn alités de niveau professionnel telles que les instantanés, le clonage et la déduplication

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				des données. Les cas d'utilisation courants incluent les charges de travail informatiques hautes performances, les référentiels de contenu et les applications nécessitant POSIX-compliant stockage partagé. Pour plus d'informations, voir Montage d'Amazon FSx pour NetApp Systèmes de fichiers ONTAP issus de conteneurs Amazon ECS. Lors du montage en volume natif de FSx pour NetApp ONTAP est disponible uniquement sur le type

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				de lancement Amazon EC2, vous pouvez accéder à votre FSx pour NetApp Données ONTAP provenant de tâches Fargate à l'aide de points d'accès S3 pour FSx. Cette approche permet à vos applicati ons conteneur isées de lire et d'écrire des données FSx pour ONTAP via l'API Amazon Simple Storage Service sans nécessiter de montage NFS ou SMB direct. Pour plus d'informa tions, voir Création d'un point d'accès Amazon S3 pour FSx pour ONTAP.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Fichiers Amazon S3	Fargate, instances gérées Amazon ECS	Linux	Persistante	Amazon S3 Files est un système de fichiers hautes performances qui fournit un accès rapide et mis en cache aux données Amazon S3 via une interface de système de fichiers montable. Les volumes S3 Files permettent à vos conteneurs d'accéder directement au système de fichiers aux données stockées dans des compartiments S3. Les cas d'utilisation courants incluent l'analyse des données, la formation à l'apprentissage automatique et

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				les applications nécessitant un accès haut débit aux données S3. Pour plus d'informations, consultez Configuration des fichiers S3 pour Amazon ECS.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Volumes Docker	Amazon EC2	Windows, Linux	Persistante	Les volumes Docker sont une fonctionnalité de l'environnement d'exécution des conteneurs Docker qui permet aux conteneurs de conserver des données en montant un répertoire à partir du système de fichiers de l'hôte. Les pilotes de volume Docker (également appelés plug-ins) sont utilisés pour intégrer les volumes de conteneurs à des systèmes de stockage externes. Les volumes Docker peuvent être gérés par des pilotes tiers ou par le pilote `local` intégré.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				Les cas d'utilisation courants des volumes Docker sont notamment la mise à disposition de volumes de données persistants ou le partage de volumes à différents emplacements sur différents conteneurs d'une même instance de conteneur. Pour plus d'informations, consultez Utilisation de volumes Docker avec Amazon ECS.

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
Montages liés	Fargate, Amazon EC2, instances gérées Amazon ECS	Windows, Linux	Éphémère	Les montages par liaison consistent en un fichier ou un répertoire sur l'hôte, tel qu'une instance Amazon EC2 ou AWS Fargate, qui est monté sur un conteneur. Les cas d'utilisation courants des montages liés sont notamment le partage d'un volume d'un conteneur source avec d'autres conteneurs de la même tâche, ou le montage d'un volume hôte ou d'un volume vide dans un ou plusieurs conteneurs. Pour plus d'informations, consultez Utilisation de montages

Volume de données	Capacité prise en charge	Systèmes d'exploitation pris en charge	Persistance du stockage	Cas d'utilisation
				liés avec Amazon ECS.

Utilisation des volumes Amazon EBS avec Amazon ECS

Les volumes Amazon Elastic Block Store (Amazon EBS) offrent un stockage par blocs hautement disponible, économique, durable et performant pour les charges de travail gourmandes en données. Les volumes Amazon EBS peuvent être utilisés avec les tâches Amazon ECS pour les applications à haut débit et gourmandes en transactions. Pour plus d'informations sur les volumes Amazon EBS, consultez la section Volumes Amazon EBS dans le Guide de l'utilisateur Amazon EBS.

Les volumes Amazon EBS attachés aux tâches Amazon ECS sont gérés par Amazon ECS en votre nom. Lors du lancement d'une tâche autonome, vous pouvez fournir la configuration qui sera utilisée pour associer un volume EBS à la tâche. Lors de la création ou de la mise à jour du service, vous pouvez fournir la configuration qui sera utilisée pour associer un volume EBS par tâche à chaque tâche gérée par le service Amazon ECS. Vous pouvez soit configurer de nouveaux volumes vides pour les attachements, soit utiliser des instantanés pour charger des données à partir de volumes existants.

> **Note**
>
> Lorsque vous utilisez des instantanés pour configurer des volumes, vous pouvez spécifier `volumeInitializationRate`, dans MiB/s, au cours de laquelle les données sont extraites de l'instantané pour créer des volumes entièrement initialisés dans un laps de temps prévisible. Pour plus d'informations sur l'initialisation des volumes, consultez la section Initialisation des volumes Amazon EBS dans le Guide de l'utilisateur Amazon EBS. Pour plus d'informations sur la configuration des volumes Amazon EBS, consultez les sections Report de la configuration du volume au moment du lancement dans une définition de tâche Amazon ECS et Spécification de la configuration du volume Amazon EBS lors du déploiement Amazon ECS.

La configuration du volume est reportée à l'heure de lancement à l'aide du paramètre `configuredAtLaunch` figurant dans la définition de tâche. En fournissant la configuration des volumes au moment du lancement plutôt que dans la définition de tâche, vous pouvez créer des définitions de tâche qui ne sont pas limitées à un type de volume de données ou à des paramètres de volume EBS spécifiques. Vous pouvez ensuite réutiliser vos définitions de tâches dans différents environnements d'exécution. Par exemple, vous pouvez fournir un débit supérieur lors du déploiement de vos charges de travail de production par rapport à vos environnements de préproduction.

Les volumes Amazon EBS associés à des tâches peuvent être chiffrés avec AWS Key Management Service (AWS KMS) clés pour protéger vos données. Pour plus d'informations, voir Chiffrement des données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS.

Pour surveiller les performances de votre volume, vous pouvez également utiliser Amazon CloudWatch métriques. Pour plus d'informations sur les métriques Amazon ECS pour les volumes Amazon EBS, consultez les sections Amazon ECS CloudWatch indicateurs et Métriques Amazon ECS Container Insights.

L'association d'un volume Amazon EBS à une tâche est prise en charge dans toutes les Régions AWS commerciales et chinoises qui prennent en charge Amazon ECS.

Systèmes d'exploitation et capacité pris en charge

Le tableau suivant indique les systèmes d'exploitation et les configurations de capacité pris en charge.

Capacity	Linux	Windows
Fargate	Les volumes Amazon EBS sont pris en charge sur la version de plateforme `1.4.0` ou ultérieure (Linux). Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.	Non pris en charge
EC2	Les volumes Amazon EBS sont pris en charge pour les tâches hébergées sur Nitroinst	Tâches hébergées sur Nitroinstances basées sur Amazon ECS-optimized

Capacity	Linux	Windows
	ances basées sur Amazon ECS-optimized Amazon Machine Images (AMI). Pour plus d'informations sur les types d'instances, consultez la section Types d'instances dans le Guide de l'utilisateur Amazon EC2. Les volumes Amazon EBS sont pris en charge sur ECS-optimized AMI `20231219` ou plus tard. Pour plus d'informations, voir Récupération d'Amazon ECS-Optimized Métadonnées AMI.	Amazon Machine Images (AMI). Pour plus d'informations sur les types d'instances, consultez la section Types d'instances dans le Guide de l'utilisateur Amazon EC2. Les volumes Amazon EBS sont pris en charge sur ECS-optimized AMI `20241017` ou plus tard. Pour plus d'informations, voir Récupération d'Amazon ECS-Optimized Métadonnées de l'AMI Windows.
Instances gérées Amazon ECS	Les volumes Amazon EBS sont pris en charge pour les tâches hébergées sur des instances gérées Amazon ECS sous Linux.	Non pris en charge

Considérations

Tenez compte des éléments suivants lorsque vous utilisez des volumes Amazon EBS :

- Vous ne pouvez pas configurer de volumes Amazon EBS à rattacher à des tâches Fargate Amazon ECS dans la zone de disponibilité `use1-az3`.
- Le type de volume Amazon EBS magnétique (`standard`) n'est pas pris en charge pour les tâches hébergées sur Fargate. Pour plus d'informations sur les types de volumes Amazon EBS, consultez la section Volumes Amazon EBS dans le Guide de l'utilisateur Amazon EC2.
- Un rôle IAM d'infrastructure Amazon ECS est requis lors de la création d'un service ou d'une tâche autonome qui configure un volume lors du déploiement. Vous pouvez associer la politique IAM `AmazonECSInfrastructureRolePolicyForVolumes` gérée par AWS au rôle, ou vous

pouvez utiliser la politique gérée comme guide pour créer et associer votre propre politique avec des autorisations qui répondent à vos besoins spécifiques. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.

- Vous pouvez associer au maximum un volume Amazon EBS à chaque tâche Amazon ECS, et il doit s'agir d'un nouveau volume. Vous ne pouvez pas associer un volume Amazon EBS existant à une tâche. Toutefois, vous pouvez configurer un nouveau volume Amazon EBS à l'aide de l'instantané d'un volume existant lors du déploiement.
- Pour utiliser les volumes Amazon EBS avec les services Amazon ECS, le contrôleur de déploiement doit être `ECS`. À la fois roulant et blue/green les stratégies de déploiement sont prises en charge lors de l'utilisation de ce contrôleur de déploiement.
- Pour qu'un conteneur de votre tâche puisse écrire sur le volume Amazon EBS monté, il doit disposer des autorisations de système de fichiers appropriées. Lorsque vous spécifiez un utilisateur non root dans votre définition de conteneur, Amazon ECS configure automatiquement le volume avec des autorisations basées sur les groupes qui permettent à l'utilisateur spécifié de lire et d'écrire sur le volume. Si aucun utilisateur n'est spécifié, le conteneur s'exécute en tant que superutilisateur et dispose d'un accès complet au volume.
- Amazon ECS ajoute automatiquement les balises réservées `AmazonECSCreated` et `AmazonECSManaged` au volume attaché. Si vous supprimez ces balises du volume, Amazon ECS ne sera pas en mesure de gérer le volume en votre nom. Pour plus d'informations sur la création d'es volumes Amazon EBS, consultez la section Étiquetage d'un volume Amazon EBS. Pour plus d'informations sur l'étiquetage de vos ressources Amazon ECS, consultez Étiquetage de vos ressources Amazon EC2.
- L'allocation de volumes à partir d'un instantané d'un volume Amazon EBS contenant des partitions n'est pas prise en charge.
- Les volumes attachés à des tâches gérées par un service ne sont pas conservés et sont toujours supprimés à la fin de la tâche.
- Vous ne pouvez pas configurer de volumes Amazon EBS à rattacher à des tâches Amazon ECS en cours d'exécution sur AWS Outposts.

Non-root comportement des utilisateurs

Lorsque vous spécifiez un utilisateur non root dans votre définition de conteneur, Amazon ECS configure automatiquement le volume Amazon EBS avec des autorisations basées sur les groupes qui permettent à l'utilisateur spécifié de lire et d'écrire sur le volume. Le volume est monté avec les caractéristiques suivantes :

- Le volume appartient à l'utilisateur root et au groupe root.
- Les autorisations de groupe sont définies pour autoriser l'accès en lecture et en écriture.
- L'utilisateur non root est ajouté au groupe approprié pour accéder au volume.

Suivez ces bonnes pratiques lorsque vous utilisez des volumes Amazon EBS avec des conteneurs non root :

- Utilisez des ID utilisateur (UID) et des ID de groupe (GID) cohérents sur les images de vos conteneurs afin de garantir des autorisations cohérentes.
- Pre-create montez des répertoires de points dans l'image de votre conteneur et définissez les droits de propriété et les autorisations appropriés.
- Testez vos conteneurs avec les volumes Amazon EBS dans un environnement de développement pour vous assurer que les autorisations du système de fichiers fonctionnent comme prévu.
- Si plusieurs conteneurs d'une même tâche partagent un volume, assurez-vous qu'ils utilisent soit des conteneurs compatibles UIDs/GIDs ou montez le volume avec des attentes d'accès cohérentes.

Report de la configuration du volume au moment du lancement dans une définition de tâche Amazon ECS

Pour configurer un volume Amazon EBS à rattacher à votre tâche, vous devez spécifier la configuration du point de montage dans la définition de tâche et nommer le volume. Vous devez également définir `configuredAtLaunch` sur `true`, car les volumes Amazon EBS ne peuvent pas être configurés pour être attaché dans la définition de tâche. Au lieu de cela, les volumes Amazon EBS sont configurés pour être attachés lors du déploiement.

Pour enregistrer la définition de tâche à l'aide du AWS Command Line Interface (AWS CLI), enregistrez le modèle en tant que fichier JSON, puis transmettez le fichier en tant qu'entrée pour le `register-task-definition` commande.

Pour créer et enregistrer une définition de tâche à l'aide du Console de gestion AWS, voir Création d'une définition de tâche Amazon ECS à l'aide de la console.

La définition de tâche suivante montre la syntaxe des objets `mountPoints` et `volumes` dans la définition de tâche. Pour plus d'informations sur les paramètres de définition des tâches, consultez la section Paramètres de définition de tâche Amazon ECS pour Fargate. Pour utiliser cet exemple, remplacez les *`user input placeholders`* par vos propres informations.

Linux

```
{
    "family": "mytaskdef",
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "networkMode": "awsvpc",
           "portMappings": [
                {
                    "name": "nginx-80-tcp",
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp",
                    "appProtocol": "http"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "/mount/ebs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

Windows

```
{
    "family": "mytaskdef",
     "memory": "4096",
     "cpu": "2048",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["EC2"]
    "containerDefinitions": [
        {
             "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<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>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 443,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEBSVolume",
                    "containerPath": "drive:\ebs",
                    "readOnly": true
                }
            ]
        }
    ],
```

```
    "volumes": [
        {
            "name": "myEBSVolume",
            "configuredAtLaunch": true
        }
    ],
    "requiresCompatibilities": [
        "FARGATE", "EC2"
    ],
    "cpu": "1024",
    "memory": "3072",
    "networkMode": "awsvpc"
}
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur des instances EC2 exécutant le système d'exploitation Windows, laissez la valeur `false` par défaut.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union (`-`) et traits de soulignement (`_`) sont autorisés. Ce nom est référencé dans le paramètre `sourceVolume` de l'objet `mountPoints` de définition du conteneur.

`configuredAtLaunch`

Type : Boolean

Obligatoire : oui, lorsque vous souhaitez attacher un volume EBS directement à une tâche.

Spécifie si un volume est configurable au lancement. Lorsque ce paramètre est défini sur `true`, vous pouvez configurer le volume lorsque vous exécutez une tâche autonome, ou lorsque vous créez ou mettez à jour un service. Lorsque cette option est définie sur `false`, vous ne pourrez pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être fourni et défini sur `true` pour configurer un volume Amazon EBS à rattacher à une tâche.

Chiffrement des données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS

Vous pouvez utiliser AWS Key Management Service (AWS KMS) pour créer et gérer des clés cryptographiques qui protègent vos données. Les volumes Amazon EBS sont chiffrés au repos à l'aide de AWS KMS keys. Les types de données suivants font l'objet d'un chiffrement :

- Données stockées au repos sur le volume
- Disque I/O

- Instantanés créés à partir du volume
- Nouveaux volumes créés à partir d'instantanés chiffrés

Les volumes Amazon EBS attachés à des tâches peuvent être chiffrés en utilisant soit une Clé gérée par AWS par défaut avec l'alias `alias/aws/ebs`, soit une clé symétrique gérée par le client spécifiée dans la configuration du volume. Par défaut Clés gérées par AWS sont uniques à chacun Compte AWS par Région AWS et sont créés automatiquement. Pour créer une clé symétrique gérée par le client, suivez les étapes décrites dans la section Création de clés KMS de chiffrement symétrique dans le Guide du développeur AWS KMS .

Vous pouvez configurer le chiffrement Amazon EBS par défaut afin que tous les nouveaux volumes soient créés et associés à une tâche dans un Région AWS sont chiffrés à l'aide de la clé KMS que vous avez spécifiée pour votre compte. Pour plus d'informations sur le chiffrement Amazon EBS et le chiffrement par défaut, consultez la section Chiffrement Amazon EBS dans le Guide de l'utilisateur Amazon EBS.

Comportement des instances gérées Amazon ECS

Vous chiffrez les volumes Amazon EBS en activant le chiffrement, soit en utilisant le chiffrement par défaut, soit en activant le chiffrement lorsque vous créez un volume que vous souhaitez chiffrer. Pour plus d'informations sur la manière d'activer le chiffrement par défaut (au niveau du compte), consultez la section Chiffrement par défaut dans le Guide de l'utilisateur Amazon EBS.

Vous pouvez configurer n'importe quelle combinaison de ces clés. L'ordre de priorité des clés KMS est le suivant :

1. La clé KMS spécifiée dans la configuration du volume. Lorsque vous spécifiez une clé KMS dans la configuration du volume, elle remplace la clé par défaut d'Amazon EBS et toute clé KMS spécifiée au niveau du compte.
2. La clé KMS spécifiée au niveau du compte. Lorsque vous spécifiez une clé KMS pour le chiffrement au niveau du cluster du stockage géré Amazon ECS, celle-ci remplace le chiffrement par défaut Amazon EBS, mais ne remplace aucune clé KMS spécifiée dans la configuration du volume.
3. Chiffrement par défaut Amazon EBS. Le chiffrement par défaut s'applique lorsque vous ne spécifiez ni clé KMS au niveau du compte ni aucune clé dans la configuration du volume. Si vous activez le chiffrement Amazon EBS par défaut, il s'agit de la clé KMS que vous spécifiez pour le chiffrement par défaut. Dans le cas contraire, la valeur par défaut est Clé gérée par AWS avec l'alias `alias/aws/ebs`.

> **Note**
>
> Si vous définissez `encrypted` sur `false` dans la configuration de votre volume, que vous ne spécifiez aucune clé KMS au niveau du compte et que vous activez le chiffrement Amazon EBS par défaut, le volume sera toujours chiffré avec la clé spécifiée pour le chiffrement Amazon EBS par défaut.

Non-Amazon Comportement des instances gérées ECS

Vous pouvez également configurer le chiffrement au niveau du cluster Amazon ECS pour le stockage géré Amazon ECS lorsque vous créez ou mettez à jour un cluster. Cluster-level le chiffrement prend effet au niveau de la tâche et peut être utilisé pour chiffrer les volumes Amazon EBS attachés à chaque tâche exécutée dans un cluster spécifique à l'aide de la clé KMS spécifiée. Pour plus d'informations sur la configuration du chiffrement au niveau du cluster pour chaque tâche, consultez ManagedStorageConfiguration dans le Référence de l'API Amazon ECS.

Vous pouvez configurer n’importe quelle combinaison de ces clés. L’ordre de priorité des clés KMS est le suivant :

1. La clé KMS spécifiée dans la configuration du volume. Lorsque vous spécifiez une clé KMS dans la configuration du volume, elle remplace la clé par défaut d’Amazon EBS et toute clé KMS spécifiée au niveau du cluster.
2. La clé KMS spécifiée au niveau du cluster. Lorsque vous spécifiez une clé KMS pour le chiffrement au niveau du cluster du stockage géré Amazon ECS, celle-ci remplace le chiffrement par défaut Amazon EBS, mais ne remplace aucune clé KMS spécifiée dans la configuration du volume.
3. Chiffrement par défaut Amazon EBS. Le chiffrement par défaut s’applique lorsque vous ne spécifiez ni clé KMS au niveau du cluster, ni clé dans la configuration du volume. Si vous activez le chiffrement Amazon EBS par défaut, il s’agit de la clé KMS que vous spécifiez pour le chiffrement par défaut. Sinon, la valeur par défaut est Clé gérée par AWS avec le pseudonyme `alias/aws/ebs`.

> **Note**
>
> Si vous définissez `encrypted` sur `false` dans la configuration de votre volume, que vous ne spécifiez aucune clé KMS au niveau du cluster et que vous activez le chiffrement

Amazon EBS par défaut, le volume sera toujours chiffré avec la clé spécifiée pour le chiffrement Amazon EBS par défaut.

Stratégie de clé KMS gérée par le client

Pour chiffrer un volume EBS attaché à votre tâche à l'aide d'une clé gérée par le client, vous devez configurer votre stratégie de clé KMS afin de vous assurer que le rôle IAM que vous utilisez pour la configuration du volume dispose des autorisations nécessaires pour utiliser la clé. La stratégie de clés KMS doit inclure à la fois les autorisations `kms:CreateGrant` et `kms:GenerateDataKey*`. Les autorisations `kms:ReEncryptTo` et `kms:ReEncryptFrom` sont nécessaires pour chiffrer les volumes créés à l'aide d'instantanés. Si vous souhaitez configurer et chiffrer uniquement les nouveaux volumes vides à attacher, vous pouvez exclure les autorisations `kms:ReEncryptTo` et `kms:ReEncryptFrom`.

L'extrait JSON suivant présente les instructions de stratégie de clé que vous pouvez attacher à votre stratégie de clé KMS. L'utilisation de ces instructions permettra à Amazon ECS d'utiliser la clé pour chiffrer le volume EBS. Pour utiliser les exemples d'instructions de politique, remplacez les *`user input placeholders`* informations par les vôtres. Comme toujours, ne configurez que les autorisations dont vous avez besoin.

```
{
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:DescribeKey",
      "Resource":"*"
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": [
      "kms:GenerateDataKey*",
      "kms:ReEncryptTo",
      "kms:ReEncryptFrom"
      ],
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
```

```
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        }
      }
    },
    {
      "Effect": "Allow",
      "Principal": { "AWS": "arn:aws:iam::111122223333:role/ecsInfrastructureRole" },
      "Action": "kms:CreateGrant",
      "Resource":"*",
      "Condition": {
        "StringEquals": {
          "kms:CallerAccount": "aws_account_id",
          "kms:ViaService": "ec2.region.amazonaws.com"
        },
        "ForAnyValue:StringEquals": {
          "kms:EncryptionContextKeys": "aws:ebs:id"
        },
        "Bool": {
          "kms:GrantIsForAWSResource": true
        }
      }
    }
```

Pour plus d'informations sur les politiques et autorisations clés, consultez les sections Stratégies de clés AWS KMS et Autorisations AWS KMS dans le Guide du développeur AWS KMS . Pour résoudre les problèmes de connexion aux volumes EBS liés aux autorisations clés, consultez la section Résolution des problèmes liés à l'attachement de volumes Amazon EBS aux tâches Amazon ECS.

Spécification de la configuration du volume Amazon EBS lors du déploiement Amazon ECS

Après avoir enregistré une définition de tâche dont le paramètre `configuredAtLaunch` est défini sur `true`, vous pouvez configurer un volume Amazon EBS lors du déploiement, lorsque vous exécutez une tâche autonome, ou lorsque vous créez ou mettez à jour un service. Pour plus d'informations sur le report de la configuration du volume au moment du lancement à l'aide du paramètre `configuredAtLaunch`, consultez la section Report de la configuration du volume au moment du lancement dans une définition de tâche Amazon ECS.

Pour configurer un volume, vous pouvez utiliser les API Amazon ECS ou transmettre un fichier JSON en entrée pour les éléments suivants AWS CLI commandes :

- `run-task` pour exécuter une tâche ECS autonome.

- `start-task` pour exécuter une tâche ECS autonome dans une instance de conteneur spécifique. Cette commande ne s'applique pas aux tâches Fargate.
- `create-service` pour créer un service ECS.
- `update-service` pour mettre à jour un service existant.

> **Note**
>
> Pour qu'un conteneur de votre tâche puisse écrire sur le volume Amazon EBS monté, il doit disposer des autorisations de système de fichiers appropriées. Lorsque vous spécifiez un utilisateur non root dans votre définition de conteneur, Amazon ECS configure automatiquement le volume avec des autorisations basées sur les groupes qui permettent à l'utilisateur spécifié de lire et d'écrire sur le volume. Si aucun utilisateur n'est spécifié, le conteneur s'exécute en tant que superutilisateur et dispose d'un accès complet au volume.

Vous pouvez également configurer un volume Amazon EBS à l'aide de l' Console de gestion AWS. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS, Création d'un déploiement de mise à jour propagée Amazon ECS et Mettre à jour un service Amazon ECS.

L'extrait de code JSON suivant présente tous les paramètres d'un volume Amazon EBS pouvant être configurés lors du déploiement. Pour utiliser ces paramètres pour configurer des volumes, remplacez *`user input placeholders`* par vos propres informations. Pour plus d'informations sur ces paramètres, consultez la section Configurations de volume.

```
"volumeConfigurations": [
        {
            "name": "ebs-volume",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "arn:aws:kms:us-
east-1:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab",
                "volumeType": "gp3",
                "sizeInGiB": 10,
                "snapshotId": "snap-12345",
                "volumeInitializationRate":100,
                "iops": 3000,
                "throughput": 125,
                "tagSpecifications": [
```

```
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                 "terminationPolicy": {
                    "deleteOnTermination": true//can't be configured for service-
managed tasks, always true
                },
                "filesystemType": "ext4"
            }
        }
    ]
```

> **Important**
>
> Assurez-vous que le `volumeName` que vous spécifiez dans la configuration est identique au `volumeName` que vous spécifiez dans votre définition de tâche.

Pour plus d'informations sur la vérification de l'état de connexion du volume, consultez la section Résolution des problèmes liés à l'attachement de volumes Amazon EBS aux tâches Amazon ECS. Pour plus d'informations sur l'infrastructure Amazon ECS Gestion des identités et des accès AWS rôle (IAM) nécessaire pour la fixation d'un volume EBS, voir Rôle IAM d'infrastructure Amazon ECS.

Voici des exemples d'extraits de code JSON illustrant la configuration des volumes Amazon EBS. Ces exemples peuvent être utilisés en enregistrant les extraits dans des fichiers JSON et en transmettant les fichiers en tant que paramètres (à l'aide du `--cli-input-json file://filename` paramètre) pour AWS CLI commandes. Remplacez les *user input placeholders* par vos propres informations.

Configuration d'un volume pour une tâche autonome

L'extrait de code suivant présente la syntaxe permettant de configurer des volumes Amazon EBS à rattacher à une tâche autonome. L'extrait de code JSON suivant présente la syntaxe de configuration des paramètres `volumeType`, `sizeInGiB`, `encrypted` et `kmsKeyId`. La configuration spécifiée dans le fichier JSON est utilisée pour créer et associer un volume EBS à la tâche autonome.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
   ]
}
```

Configuration d'un volume lors de la création du service

L'extrait de code suivant présente la syntaxe permettant de configurer des volumes Amazon EBS à rattacher à des tâches gérées par un service. Les volumes proviennent de l'instantané spécifié à l'aide du `snapshotId` paramètre à un taux de 200 MiB/s. La configuration spécifiée dans le fichier JSON est utilisée pour créer et attacher un volume EBS à chaque tâche gérée par le service.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": [
        {
            "name": "myEbsVolume",
            "managedEBSVolume": {
              "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
```

```
            "snapshotId": "snap-12345",
            "volumeInitializationRate": 200
        }
    }
  ]
}
```

Configuration d'un volume lors de la mise à jour d'un service

L'extrait de code JSON suivant présente la syntaxe permettant de mettre à jour un service qui ne disposait pas auparavant de volumes Amazon EBS configurés pour être rattachés à des tâches. Vous devez fournir l'ARN d'une révision de définition de tâche pour laquelle `configuredAtLaunch` est défini sur `true`. L'extrait de code JSON suivant présente la syntaxe de configuration des paramètres `volumeType`, `sizeInGiB`, `throughput`, `iops` et `filesystemType`. Cette configuration est utilisée pour créer et rattacher un volume EBS à chaque tâche gérée par le service.

```
{
  "cluster": "mycluster",
  "taskDefinition": "mytaskdef",
  "service": "mysvc",
  "desiredCount": 2,
  "volumeConfigurations": [
      {
          "name": "myEbsVolume",
          "managedEBSVolume": {
            "roleArn":"arn:aws:iam::1111222333:role/ecsInfrastructureRole",
             "volumeType": "gp3",
              "sizeInGiB": 100,
               "iops": 3000,
              "throughput": 125,
              "filesystemType": "ext4"
          }
      }
  ]
}
```

Configuration d'un service pour qu'il n'utilise plus les volumes Amazon EBS

L'extrait de code JSON suivant présente la syntaxe permettant de mettre à jour un service afin qu'il n'utilise plus les volumes Amazon EBS. Vous devez fournir l'ARN d'une définition de tâche pour laquelle `configuredAtLaunch` est défini sur `false` ou d'une définition de tâche qui ne

comporte pas le paramètre `configuredAtLaunch`. Vous devez également fournir un objet `volumeConfigurations` vide.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "service": "mysvc",
   "desiredCount": 2,
   "volumeConfigurations": []
}
```

Stratégie de résiliation pour les volumes Amazon EBS

Lorsqu'une tâche Amazon ECS est résiliée, Amazon ECS utilise la valeur `deleteOnTermination` pour déterminer si le volume Amazon EBS associé à la tâche résiliée doit être supprimé. Par défaut, les volumes EBS rattachés à des tâches sont supprimés lorsque la tâche est résiliée. Pour les tâches autonomes, vous pouvez modifier ce paramètre afin de préserver le volume à la fin de la tâche.

> **Note**
>
> Les volumes rattachés à des tâches gérées par un service ne sont pas conservés et sont toujours supprimés lorsque la tâche est résiliée.

Étiquetage des volumes Amazon EBS

Vous pouvez étiqueter les volumes Amazon EBS à l'aide de l'objet `tagSpecifications`. À l'aide de cet objet, vous pouvez fournir vos propres balises et définir la propagation des balises à partir de la définition de tâche ou de service, selon que le volume est associé à une tâche autonome ou à une tâche dans un service. Le nombre maximal de balises pouvant être attachées à un volume est 50.

> **Important**
>
> Amazon ECS attache automatiquement les balises réservées `AmazonECSCreated` et `AmazonECSManaged` à un volume Amazon EBS. Cela signifie que vous pouvez contrôler l'attachement d'un maximum de 48 balises supplémentaires à un volume. Ces balises supplémentaires peuvent être définies par l'utilisateur, ECS-managed, ou des balises propagées.

Si vous souhaitez ajouter Amazon ECS-managed tags pour votre volume, vous devez définir `enableECSManagedTags` pour `true` dans votre `UpdateService`, `CreateService`,`RunTask` ou `StartTask` appel. Si vous activez Amazon ECS-managed balises, Amazon ECS balisera automatiquement le volume avec les informations relatives au cluster et au service (`aws:ecs:`*`clusterName`* et `aws:ecs:`*`serviceName`*). Pour plus d'informations sur l'étiquetage de vos ressources Amazon ECS, consultez Étiquetage de vos ressources Amazon EC2.

L'extrait de code JSON suivant montre la syntaxe permettant d'étiqueter chaque volume Amazon EBS associé à chaque tâche d'un service avec une balise définie par l'utilisateur. Pour utiliser cet exemple pour créer un service, remplacez chaque *`user input placeholders`* par vos propres informations.

```
{
   "cluster": "mycluster",
   "taskDefinition": "mytaskdef",
   "serviceName": "mysvc",
   "desiredCount": 2,
   "enableECSManagedTags": true,
   "volumeConfigurations": [
        {
            "name": "datadir",
            "managedEBSVolume": {
                "volumeType": "gp3",
                "sizeInGiB": 100,
                 "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "key1",
                                "value": "value1"
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn":"arn:aws:iam:1111222333:role/ecsInfrastructureRole",
                "encrypted": true,
                "kmsKeyId":
 "arn:aws:kms:region:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
            }
        }
```

```
    ]
}
```

> **Important**
>
> Vous devez spécifier un type de ressource de `volume` pour étiqueter les volumes Amazon EBS.

Performances des volumes Amazon EBS pour les tâches Fargate à la demande

performances de base en termes d'IOPS et de débit des volumes Amazon EBS disponibles pour une tâche Fargate à la demande dépendent du nombre total d'unités d'UC que vous demandez pour la tâche. Si vous demandez 0,25, 0,5 ou 1 unité d'UC virtuelle (vCPU) pour votre tâche Fargate, nous vous recommandons de configurer un volume SSD à usage général (`gp2` ou `gp3`) ou un volume de disque dur (HDD) (`st1` ou `sc1`). Si vous demandez plus d'un vCPU pour votre tâche Fargate, les limites de performances de base suivantes s'appliquent à un volume Amazon EBS rattaché à la tâche. Vous pouvez obtenir des performances EBS temporairement supérieures aux limites suivantes. Cependant, nous vous recommandons de planifier votre charge de travail en fonction de ces limites.

Unités d'UC demandées (en vCPU)	IOPS Amazon EBS de référence (16 Ko) I/O)	Débit Amazon EBS de référence (en MiBps, 128 Ko I/O)	Bande passante de base (en Mbit/s)
2	3 000	75	360
4	5 000	120	1 150
8	10 000	250	2 300
16	15 000	500	4 500

> **Note**
>
> Lorsque vous configurez un volume Amazon EBS à rattacher à une tâche Fargate, la limite de performance Amazon EBS pour la tâche Fargate est partagée entre le stockage éphémère de la tâche et le volume attaché.

Performances des volumes Amazon EBS pour les tâches EC2

Amazon EBS fournit les types de volume suivants, qui ont des caractéristiques de performances et des prix différents, ce qui vous permet d'adapter vos performances de stockage et vos coûts en fonction des besoins de vos applications. Pour plus d'informations sur les performances, notamment les IOPS par volume et le débit par volume, consultez la section Types de volumes Amazon EBS dans le Guide de l'utilisateur d'Amazon Elastic Block Store.

Performances des volumes Amazon EBS pour les tâches d'instances gérées Amazon ECS

Amazon EBS fournit les types de volume suivants, qui ont des caractéristiques de performances et des prix différents, ce qui vous permet d'adapter vos performances de stockage et vos coûts en fonction des besoins de vos applications. Pour plus d'informations sur les performances, notamment les IOPS par volume et le débit par volume, consultez la section Types de volumes Amazon EBS dans le Guide de l'utilisateur d'Amazon Elastic Block Store.

Résolution des problèmes liés à l'attachement de volumes Amazon EBS aux tâches Amazon ECS

Vous devrez peut-être résoudre des problèmes ou procéder à des vérifications en lien à l'attachement des volumes Amazon EBS aux tâches Amazon ECS.

Vérification du statut de l'attachement aux volumes

Vous pouvez utiliser le Console de gestion AWS pour afficher l'état de la pièce jointe d'un volume Amazon EBS à une tâche Amazon ECS. Si la tâche démarre et que l'attachement échoue, vous verrez également s'afficher une raison de statut que vous pourrez utiliser pour la résolution des problèmes. Le volume créé sera supprimé et la tâche arrêtée. Pour plus d'informations sur les raisons du statut, consultez la section Raisons du statut de l'attachement d'un volume Amazon EBS aux tâches Amazon ECS.

Pour consulter le statut de l'attachement d'un volume et la raison du statut à l'aide de la console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Sur la page Clusters, choisissez le cluster dans lequel votre tâche est exécutée. La page des détails du cluster s'affiche.
3. Sur la page de détails du cluster, choisissez l'onglet Tâches.
4. Choisissez la tâche pour laquelle vous souhaitez afficher le statut de l'attachement au volume. Vous devrez peut-être utiliser Filtrer le statut souhaité et choisir Arrêté si la tâche que vous souhaitez examiner s'est arrêtée.
5. Sur la page de détails de la tâche, choisissez l'onglet Volumes. Vous pourrez voir le statut de l'attachement au volume Amazon EBS sous Statut de l'attachement. Si l'attachement du volume à la tâche échoue, vous pouvez sélectionner le statut sous Statut de l'attachement pour afficher la cause de l'échec.

Vous pouvez également consulter l'état des pièces jointes au volume d'une tâche et la raison du statut associée à l'aide du DescribeTasks API.

Échecs de services et de tâches

Vous pouvez rencontrer des échecs de service ou de tâche qui ne sont pas spécifiques aux volumes Amazon EBS et qui peuvent affecter l'attachement des volumes. Pour plus d'informations, veuillez consulter la rubrique

- Messages d'événement de service
- Codes d'erreur des tâches arrêtées
- Raisons de défaillance de l'API

Le conteneur ne peut pas écrire sur le volume Amazon EBS

Non-root utilisateur sans autorisations appropriées

Lorsque vous spécifiez un utilisateur non root dans votre définition de conteneur, Amazon ECS configure automatiquement le volume avec des autorisations basées sur les groupes pour autoriser l'accès en écriture. Toutefois, si vous rencontrez toujours des problèmes d'autorisation :

- Vérifiez que `user` le paramètre est correctement spécifié dans la définition de votre conteneur en utilisant le format `uid:gid` (par exemple, `1001:1001`).
- Assurez-vous que l'image de votre conteneur ne remplace pas les autorisations de l'utilisateur une fois le volume monté.

- Vérifiez que votre application s'exécute avec l'ID utilisateur attendu en examinant les journaux du conteneur ou en utilisant Amazon ECS Exec pour inspecter le conteneur en cours d'exécution.

Utilisateur root ayant des problèmes d'autorisation

Si aucun utilisateur n'est spécifié dans la définition de votre conteneur, celui-ci s'exécute en tant que superutilisateur et doit avoir un accès complet au volume. Si vous rencontrez des problèmes :

- Vérifiez que le volume est correctement monté en vérifiant les points de montage à l'intérieur du conteneur.
- Assurez-vous que le volume n'est pas configuré en lecture seule dans la configuration de votre point de montage.

Multi-container tâches avec différents utilisateurs

Dans les tâches impliquant plusieurs conteneurs exécutés en tant qu'utilisateurs différents, Amazon ECS gère automatiquement les autorisations de groupe pour permettre à tous les utilisateurs spécifiés d'écrire sur le volume. Si les conteneurs ne peuvent pas écrire :

- Vérifiez que tous les conteneurs nécessitant un accès en écriture disposent du `user` paramètre correctement configuré.
- Vérifiez que le volume est monté dans tous les conteneurs qui doivent y accéder.

Pour plus d'informations sur la configuration des utilisateurs dans les définitions de conteneurs, consultez Paramètres de définition de tâches Amazon ECS pour Fargate .

Raisons du statut de l’attachement d’un volume Amazon EBS aux tâches Amazon ECS

Utilisez la référence suivante pour résoudre les problèmes que vous pourriez rencontrer sous la forme de raisons de statut dans le Console de gestion AWS lorsque vous configurez des volumes Amazon EBS pour les associer à des tâches Amazon ECS. Pour plus d’informations sur la localisation des raisons de ces statuts dans la console, consultez la section Vérification du statut de l’attachement aux volumes.

ECS n'a pas pu assumer le rôle d'infrastructure ECS configuré « arn:aws:iam : :*111122223333*:rôle/*ecsInfrastructureRole*'. Vérifiez que le rôle transmis possède la bonne relation d’approbation avec Amazon ECS

Cette raison de statut apparaît dans les scénarios suivants.

- Vous fournissez un rôle IAM sans que la stratégie d'approbation requise ne soit attachée. Amazon ECS ne peut pas accéder au rôle IAM d'infrastructure Amazon ECS que vous fournissez si le rôle ne dispose pas de la stratégie d'approbation requise. La tâche peut rester bloquée dans l'état `DEPROVISIONING`. Pour de plus amples informations sur la stratégie d'approbation requise, consultez la section Rôle IAM d'infrastructure Amazon ECS.
- Votre utilisateur IAM n'est pas autorisé à transmettre le rôle d'infrastructure Amazon ECS à Amazon ECS. La tâche peut rester bloquée dans l'état `DEPROVISIONING`. Pour éviter ce problème, vous pouvez associer l'autorisation `PassRole` à votre utilisateur. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
- Votre rôle IAM ne dispose pas des autorisations requises pour attacher un volume Amazon EBS. La tâche peut rester bloquée dans l'état `DEPROVISIONING`. Pour plus d'informations sur les autorisations spécifiques requises pour attacher des volumes Amazon EBS à des tâches, consultez la section Rôle IAM d'infrastructure Amazon ECS.

> **Note**
>
> Ce message d'erreur peut également s'afficher en raison d'un retard dans la propagation des rôles. Si une nouvelle tentative d'utilisation du rôle après quelques minutes d'attente ne résout pas le problème, vous avez peut-être mal configuré la stratégie d'approbation pour le rôle.

ECS n'a pas réussi à configurer le volume EBS. Rencontré IdempotentParameterMismatch« ; « Le jeton client que vous avez fourni est associé à une ressource déjà supprimée. Veuillez utiliser un autre jeton client. »

Les suivants AWS KMS des scénarios clés peuvent mener à une `IdempotentParameterMismatch` message qui apparaît :

- Vous spécifiez un ARN, un ID ou un alias de clé KMS non valide. Dans ce scénario, la tâche peut sembler être lancée avec succès, mais elle finit par échouer car AWS authentifie la clé KMS de manière asynchrone. Pour plus d'informations, consultez la section Chiffrement Amazon EBS dans le Guide de l'utilisateur Amazon EC2.
- Vous fournissez une clé gérée par le client qui ne dispose pas des autorisations permettant au rôle IAM d'infrastructure Amazon ECS d'utiliser la clé pour le chiffrement. Pour éviter les problèmes d'autorisation liés aux politiques clés, consultez l'exemple AWS KMS politique clé en Chiffrement des données pour les volumes Amazon EBS.

Vous pouvez configurer Amazon EventBridge pour envoyer les événements de volume Amazon EBS et les événements de changement d'état des tâches Amazon ECS à une cible, telle qu'Amazon CloudWatch groupes. Vous pouvez ensuite utiliser ces événements pour identifier le problème spécifique lié aux clés gérées par le client qui a affecté l'attachement du volume. Pour plus d'informations, veuillez consulter la rubrique

- Comment puis-je créer un CloudWatch groupe de journaux à utiliser comme cible pour un EventBridge règle ? sur AWS Re : POST.
- Événements de modification de l'état de tâche
- Amazon EventBridge événements pour Amazon EBS dans le Guide de l'utilisateur d'Amazon EBS.

ECS a dépassé le délai imparti lors de la configuration de l'attachement du volume EBS à votre tâche.

Les scénarios suivants liés au format du système de fichiers entraînent l'affichage de ce message.

- Le format du système de fichiers que vous spécifiez lors de la configuration n'est pas compatible avec le système d'exploitation de la tâche.
- Vous configurez un volume Amazon EBS pour qu'il soit créé à partir d'un instantané, et le format du système de fichiers de l'instantané n'est pas compatible avec le système d'exploitation de la tâche. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané.

Vous pouvez utiliser les journaux de l'agent de conteneur Amazon ECS pour résoudre les problèmes liés à ce message pour les tâches EC2. Pour plus d'informations, consultez les sections Emplacements des fichiers journaux Amazon ECS et Collecteur de journaux Amazon ECS.

Utilisation des volumes Amazon EFS avec Amazon ECS

Amazon Elastic File System (Amazon EFS) fournit un stockage de fichiers simple et évolutif à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer de l'espace de stockage qui leur est nécessaire, au moment où elles en ont besoin.

Vous pouvez utiliser le système de fichiers Amazon EFS avec Amazon ECS pour exporter des données du système de fichiers dans l'ensemble de votre flotte d'instances de conteneur. Ainsi,

vos tâches ont accès au même stockage permanent, quelle que soit l'instance sur laquelle elles se retrouvent. Vos définitions de tâche doivent faire référence aux montages des volumes de l'instance de conteneur pour utiliser le système de fichiers.

Pour obtenir un didacticiel, consultez Configuration des systèmes de fichiers Amazon EFS pour Amazon ECS à l'aide de la console.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez des volumes Amazon EFS :

- Pour les tâches exécutées sur EC2, la prise en charge du système de fichiers Amazon EFS a été ajoutée en tant qu'aperçu public avec Amazon ECS-optimized Version AMI `20191212` avec la version 1.35.0 de l'agent de conteneur. Cependant, la prise en charge du système de fichiers Amazon EFS est entrée en disponibilité générale avec Amazon ECS-optimized Version AMI `20200319` avec la version 1.38.0 de l'agent de conteneur, qui contenait le point d'accès Amazon EFS et les fonctionnalités d'autorisation IAM. Nous vous recommandons d'utiliser Amazon ECS-optimized Version AMI `20200319` ou ultérieurement pour utiliser ces fonctionnalités. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

  > **Note**
  >
  > Si vous créez votre propre AMI, vous devez utiliser l'agent de conteneur version 1.38.0 ou une version ultérieure, `ecs-init` version 1.38.0-1 ou une version ultérieure et exécuter les commandes suivantes sur votre instance Amazon EC2 pour activer le plug-in de volume Amazon ECS. Les commandes varient selon que vous utilisez Amazon Linux 2 ou Amazon Linux comme image de base.
  >
  > Amazon Linux 2
  >
  > ```
  > yum install amazon-efs-utils
  > systemctl enable --now amazon-ecs-volume-plugin
  > ```
  >
  > Amazon Linux
  >
  > ```
  > yum install amazon-efs-utils
  > sudo shutdown -r now
  > ```

- Pour les tâches hébergées sur Fargate, les systèmes de fichiers Amazon EFS sont pris en charge sur la version 1.4.0 de la plateforme ou une version ultérieure (Linux). Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.
- Lorsque vous utilisez des volumes Amazon EFS pour des tâches hébergées sur Fargate, Fargate crée un conteneur de supervision chargé de la gestion du volume Amazon EFS. Le conteneur superviseur utilise une petite partie de la mémoire et de l'UC de la tâche. Le conteneur de supervision est visible lors de l'interrogation du point de terminaison de métadonnées de tâche version 4. De plus, il est visible dans CloudWatch Container Insights comme nom du conteneur `aws-fargate-supervisor`. Pour plus d'informations sur l'utilisation d'EC2, consultez la section Point de terminaison des métadonnées de tâches Amazon ECS version 4. Pour plus d'informations sur l'utilisation de Fargate, consultez la section Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate.
- L'utilisation de volumes Amazon EFS ou la spécification d'une `EFSVolumeConfiguration` n'est pas prise en charge sur les instances externes.
- L'utilisation de volumes Amazon EFS est prise en charge pour les tâches exécutées sur des instances gérées Amazon ECS.
- Nous vous recommandons de définir le paramètre `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` dans le fichier de configuration de l'agent sur une valeur inférieure à la valeur par défaut (environ une heure). Cette modification permet d'éviter l'expiration des informations d'identification de montage EFS et permet le nettoyage des montages qui ne sont pas utilisés. Pour plus d'informations, voir Configuration de l'agent de conteneur Amazon ECS.

Utilisation des points d'accès Amazon EFS

Les points d'accès Amazon EFS sont des points d'entrée spécifiques à l'application dans un système de fichiers EFS pour gérer l'accès des applications aux jeux de données partagés. Pour de plus amples informations sur les points d'accès Amazon EFS et sur la façon de les contrôler, veuillez consulter Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Les points d'accès peuvent appliquer de manière forcée une identité d'utilisateur, y compris les groupes POSIX de l'utilisateur, pour toutes les demandes de système de fichiers effectuées via le point d'accès. Les points d'accès peuvent également imposer un répertoire racine différent au système de fichiers. Cela permet aux clients d'accéder uniquement aux données stockées dans le répertoire spécifié ou dans ses sous-répertoires.

> **Note**
>
> Lors de la création d'un point d'accès EFS, spécifiez un chemin d'accès sur le système de fichiers qui servira de répertoire racine. Lorsque vous faites référence au système de fichiers EFS avec un ID de point d'accès dans votre définition de tâche Amazon ECS, le répertoire racine doit être omis ou défini sur `/` ce qui permet d'appliquer le chemin défini sur le point d'accès EFS.

Vous pouvez utiliser le rôle IAM d'une tâche Amazon ECS pour imposer l'utilisation d'un point d'accès particulier par des applications spécifiques. En combinant des politiques IAM avec des points d'accès, vous pouvez fournir un accès sécurisé à des ensembles de données spécifiques pour vos applications. Pour plus d'informations sur l'utilisation des rôles IAM de tâche, consultez rôle IAM de tâche Amazon ECS.

Pratiques exemplaires en matière d’utilisation des volumes Amazon EFS avec Amazon ECS

Tenez compte des recommandations suivantes en matière de pratique exemplaire lorsque vous utilisez Amazon EFS avec Amazon ECS.

Contrôles de sécurité et d’accès pour les volumes Amazon EFS

Amazon EFS propose des fonctionnalités de contrôle d’accès que vous pouvez utiliser pour garantir que les données stockées dans un système de fichiers Amazon EFS sont sécurisées et accessibles uniquement depuis les applications qui en ont besoin. Vous pouvez sécuriser les données en activant le chiffrement au repos et en transit. Pour en savoir plus, consultez Chiffrement des données dans Amazon EFS que vous trouverez dans le guide de l’utilisateur Amazon Elastic File System.

Outre le chiffrement des données, vous pouvez également utiliser Amazon EFS pour restreindre l’accès à un système de fichiers. Vous pouvez implémenter le contrôle d’accès dans EFS de trois manières.

- Groupes de sécurité : avec les cibles de montage Amazon EFS, vous pouvez configurer un groupe de sécurité utilisé pour autoriser ou refuser le trafic réseau. Vous pouvez configurer le groupe de sécurité associé à Amazon EFS afin d’autoriser le trafic NFS (port 2049) provenant du groupe de sécurité associé à vos instances Amazon ECS ou, lorsque vous utilisez le mode réseau `awsvpc`, à la tâche Amazon ECS.
- IAM : vous pouvez restreindre l’accès à un système de fichiers Amazon EFS à l’aide d’IAM. Une fois configurées, les tâches Amazon ECS nécessitent un rôle IAM pour accéder au système de

fichiers afin de monter un système de fichiers EFS. Pour plus d'informations, consultez la section Utilisation d'IAM pour contrôler l'accès aux données du système de fichiers dans le Guide de l'utilisateur Amazon Elastic File System.

Les politiques IAM peuvent également appliquer des conditions prédéfinies, telles que l'obligation pour un client d'utiliser TLS lors de la connexion à un système de fichiers Amazon EFS. Pour plus d'informations, consultez la section Clés de condition Amazon EFS pour les clients dans le Guide de l'utilisateur Amazon Elastic File System.

- Points d'accès Amazon EFS : les points d'accès Amazon EFS sont des points d'entrée spécifiques à l'application dans un système de fichiers Amazon EFS. Vous pouvez utiliser des points d'accès pour imposer une identité utilisateur, y compris les groupes POSIX de l'utilisateur, pour toutes les requêtes du système de fichiers effectuées via le point d'accès. Les points d'accès peuvent également imposer un répertoire racine différent au système de fichiers. Cela permet aux clients d'accéder uniquement aux données stockées dans le répertoire spécifié ou dans ses sous-répertoires.

politiques IAM

Vous pouvez utiliser les politiques IAM pour contrôler l'accès au système de fichiers Amazon EFS.

Vous pouvez spécifier les actions suivantes pour les clients sur un système de fichiers à l'aide d'une stratégie de système de fichiers.

Action	Description
`elasticfilesystem:ClientMount`	Fournit un accès en lecture seule à un système de fichiers.
`elasticfilesystem:ClientWrite`	Fournit les droits d'écriture sur un système de fichiers.
`elasticfilesystem:ClientRootAccess`	Permet d'utiliser l'utilisateur root lors de l'accès à un système de fichiers.

Vous devez spécifier chaque action dans une politique. Les politiques peuvent être définies comme suit :

- Client-based - Associez la politique au rôle de la tâche

Définissez l'option Autorisation IAM lorsque vous créez la définition de tâche.

- Resource-based - Associez la politique au système de fichiers Amazon EFS

  Si la stratégie basée sur les ressources n'existe pas, l'accès est accordé par défaut à tous les principaux (*) lors de la création du système de fichiers.

Lorsque vous définissez l'option Autorisation IAM, nous fusionnons la politique associée au rôle de tâche et la politique basée sur les ressources Amazon EFS. L'option Autorisation IAM transmet l'identité de la tâche (le rôle de la tâche) avec la politique à Amazon EFS. Cela permet à la politique basée sur les ressources Amazon EFS d'avoir un contexte pour l'utilisateur ou le rôle IAM spécifié dans la politique. Si vous ne définissez pas cette option, la politique au niveau des ressources Amazon EFS identifie l'utilisateur IAM comme « anonyme ».

Envisagez d'implémenter les trois contrôles d'accès sur un système de fichiers Amazon EFS pour une sécurité maximale. Par exemple, vous pouvez configurer le groupe de sécurité attaché à un point de montage Amazon EFS pour n'autoriser que le trafic NFS entrant provenant d'un groupe de sécurité associé à votre instance de conteneur ou à votre tâche Amazon ECS. En outre, vous pouvez configurer Amazon EFS pour exiger un rôle IAM pour accéder au système de fichiers, même si la connexion provient d'un groupe de sécurité autorisé. Enfin, vous pouvez utiliser les points d'accès Amazon EFS pour imposer les autorisations des utilisateurs POSIX et spécifier des répertoires racines pour les applications.

L'extrait de définition de tâche suivant illustre comment monter un système de fichiers Amazon EFS à l'aide d'un point d'accès.

```
"volumes": [
    {
      "efsVolumeConfiguration": {
        "fileSystemId": "fs-1234",
        "authorizationConfig": {
          "accessPointId": "fsap-1234",
          "iam": "ENABLED"
        },
        "transitEncryption": "ENABLED",
        "rootDirectory": ""
      },
      "name": "my-filesystem"
    }
]
```

Performance des volumes Amazon EFS

Amazon EFS propose deux modes de performance : General Purpose et Max I/O. General Purpose convient aux applications sensibles à la latence telles que les systèmes de gestion de contenu et CI/CD outils. En revanche, Max I/O les systèmes de fichiers sont adaptés aux charges de travail telles que l'analyse des données, le traitement multimédia et l'apprentissage automatique. Ces charges de travail doivent effectuer des opérations parallèles à partir de centaines, voire de milliers de conteneurs, et nécessitent un débit global et un nombre d'opérations d'entrée/sortie par seconde (IOPS) aussi élevés que possible. Pour plus d'informations, consultez la section Modes de performances Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Certaines charges de travail sensibles à la latence nécessitent à la fois la plus élevée I/O niveaux fournis par Max I/O le mode performance et la latence plus faible qui sont fournis par le mode de performance à usage général. Pour ce type de charge de travail, nous vous recommandons de créer plusieurs systèmes de fichiers en mode de performance Usage général. De cette manière, vous pouvez répartir la charge de travail de vos applications sur tous ces systèmes de fichiers, à condition que la charge de travail et les applications le permettent.

Débit des volumes Amazon EFS

Tous les systèmes de fichiers Amazon EFS présentent un débit mesuré associé qui est déterminé soit par la quantité de débit alloué pour les systèmes de fichiers utilisant le débit alloué, soit par la quantité de données stockées dans la classe de stockage EFS Standard ou One Zone pour les systèmes de fichiers utilisant le débit de débordement. Pour plus d'informations, consultez la section Compréhension du débit mesuré dans le Guide de l'utilisateur Amazon Elastic File System.

Le mode de débit par défaut pour les systèmes de fichiers Amazon EFS est le mode de débordement. Avec le mode de débordement, le débit disponible pour un système de fichiers augmente ou diminue au fur et à mesure que le système de fichiers grandit. Étant donné que les charges de travail basées sur des fichiers connaissent généralement des pics, nécessitant des niveaux de débit élevés pendant certaines périodes et des niveaux de débit plus faibles le reste du temps, Amazon EFS est conçu pour déborder afin d'autoriser des niveaux de débit élevés pendant certaines périodes. En outre, étant donné que de nombreuses charges de travail sont gourmandes en lecture, les opérations de lecture sont mesurées selon un ratio de 1:3 par rapport aux autres opérations NFS (comme l'écriture).

Tous les systèmes de fichiers Amazon EFS offrent des performances de base constantes de 50 %. MB/s pour chaque To de stockage Amazon EFS Standard ou Amazon EFS One Zone. Tous les systèmes de fichiers (quelle que soit leur taille) peuvent atteindre 100 MB/s. Les systèmes de fichiers

dotés de plus de 1 To de stockage EFS Standard ou EFS One Zone peuvent atteindre 100 MB/s pour chaque TB. Comme les opérations de lecture sont mesurées selon un ratio de 1:3, vous pouvez conduire jusqu'à 300 MiBs/s pour chaque TiB de débit de lecture. Lorsque vous ajoutez des données à votre système de fichiers, le débit maximal disponible pour le système de fichiers évolue de manière linéaire et automatique en fonction de votre stockage dans la classe de stockage Amazon EFS Standard. Si vous avez besoin d'un débit supérieur à celui que vous pouvez obtenir avec la quantité de données stockées, vous pouvez configurer le débit alloué en fonction des besoins spécifiques de votre charge de travail.

Le débit du système de fichiers est partagé entre toutes les instances Amazon EC2 connectées à un système de fichiers. Par exemple, un système de fichiers de 1 To pouvant atteindre 100 MB/s du débit peut en piloter 100 MB/s à partir d'une seule instance Amazon EC2, chaque instance peut en gérer 10 MB/s. Pour plus d'informations, consultez la section Performances Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Optimisation des coûts pour les volumes Amazon EFS

Amazon EFS simplifie la mise à l'échelle du stockage pour vous. Les systèmes de fichiers Amazon EFS augmentent automatiquement au fur et à mesure que vous ajoutez des données. En particulier avec le mode de débit de débordement Amazon EFS, le débit sur Amazon EFS est mis à l'échelle à mesure que la taille de votre système de fichiers dans la classe de stockage standard augmente. Pour améliorer le débit sans payer de frais supplémentaires pour le débit alloué sur un système de fichiers EFS, vous pouvez partager un système de fichiers Amazon EFS avec plusieurs applications. À l'aide des points d'accès Amazon EFS, vous pouvez implémenter l'isolation du stockage dans les systèmes de fichiers Amazon EFS partagés. Ainsi, même si les applications partagent toujours le même système de fichiers, elles ne peuvent accéder aux données que si vous les y autorisez.

Au fur et à mesure que vos données augmentent, Amazon EFS vous aide à déplacer automatiquement les fichiers rarement consultés vers une classe de stockage inférieure. L'Amazon EFS Standard-Infrequent La classe de stockage Access (IA) réduit les coûts de stockage pour les fichiers qui ne sont pas consultés tous les jours. La haute disponibilité, la durabilité, l'élasticité et l'accès au système de fichiers POSIX fournis par Amazon EFS sont conservés. Pour plus d'informations, consultez la section Classes de stockage EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Envisagez d'utiliser les stratégies de cycle de vie Amazon EFS pour réaliser automatiquement des économies en transférant les fichiers rarement consultés vers le stockage Amazon EFS IA. Pour plus d'informations, consultez Gestion du cycle de vie Amazon EFS dans le guide de l'utilisateur Amazon Elastic File System.

Lorsque vous créez un système de fichiers Amazon EFS, vous pouvez choisir si Amazon EFS réplique vos données sur plusieurs zones de disponibilité (standard) ou les stocke de manière redondante dans une seule zone de disponibilité. La classe de stockage Amazon EFS One Zone permet de réduire considérablement les coûts de stockage par rapport aux classes de stockage Amazon EFS Standard. Envisagez d'utiliser la classe de stockage Amazon EFS One Zone pour les charges de travail ne nécessitant pas de résilience multi-AZ. Vous pouvez encore réduire le coût du stockage Amazon EFS One Zone en transférant les fichiers rarement consultés vers Amazon EFS One Zone-Infrequent Accès. Pour plus d'informations, veuillez consulter Amazon EFS Infrequent Access.

Protection des données des volumes Amazon EFS

Amazon EFS stocke vos données de manière redondante dans plusieurs zones de disponibilité pour les systèmes de fichiers à l'aide de classes de stockage standard. Si vous sélectionnez les classes de stockage Amazon EFS One Zone, vos données sont stockées de manière redondante dans une seule zone de disponibilité. Amazon EFS est en outre conçu pour fournir 99,99999999999 % (11 9) de durabilité sur une année donnée.

Comme dans tout environnement, il est recommandé de disposer d'une sauvegarde et de mettre en place des mesures de protection contre les suppressions accidentelles. Pour les données Amazon EFS, cette meilleure pratique inclut une sauvegarde fonctionnelle et régulièrement testée à l'aide de AWS Backup. Les systèmes de fichiers utilisant les classes de stockage Amazon EFS One Zone sont configurés par défaut pour sauvegarder automatiquement les fichiers lors de la création du système de fichiers, sauf si vous choisissez de désactiver cette fonctionnalité. Pour plus d'informations, consultez la section Sauvegarde des systèmes de fichiers EFS dans le Guide de l'utilisateur Amazon Elastic File System.

Spécification d'un système de fichiers Amazon EFS dans une définition de tâche Amazon ECS

Pour utiliser des volumes de système de fichiers Amazon EFS pour vos conteneurs, vous devez spécifier les configurations de volume et de point de montage dans votre définition de tâche. L'extrait de JSON de définition de tâche indiqué ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur.

```
{
    "containerDefinitions": [
        {
            "name": "container-using-efs",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
```

```
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": [
                "ls -la /mount/efs"
            ],
            "mountPoints": [
                {
                    "sourceVolume": "myEfsVolume",
                    "containerPath": "/mount/efs",
                    "readOnly": true
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "myEfsVolume",
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1234",
                "rootDirectory": "/path/to/my/data",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": integer,
                "authorizationConfig": {
                    "accessPointId": "fsap-1234",
                    "iam": "ENABLED"
                }
            }
        }
    ]
}
```

`efsVolumeConfiguration`

Type : objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans `authorizationConfig`, le paramètre de répertoire racine doit être omis ou défini sur `/` ce qui permet d'appliquer le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : Integer

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, il utilise la stratégie de

sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Type : objet

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d'accès est spécifié, la valeur du répertoire racine dans `efsVolumeConfiguration` doit être omise ou définie sur `/` qui applique le chemin défini sur le point d'accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique s'il faut ou non utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

Configuration des systèmes de fichiers Amazon EFS pour Amazon ECS à l’aide de la console

Découvrez comment utiliser les systèmes de fichiers Amazon Elastic File System (Amazon EFS) avec Amazon ECS.

Étape 1 : Créer un cluster Amazon ECS

Pour créer un cluster Amazon ECS, effectuez les étapes suivantes.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour plus d'informations, consultez the section called "Exemples de cluster Amazon ECS".

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Cluster configuration (Configuration de cluster), dans Cluster name (Nom du cluster), saisissez `EFS-tutorial`.
6. (Facultatif) Pour modifier le VPC et les sous-réseaux d'où vos tâches et services se lancent, sous Networking (Réseaux), effectuez l'une des opérations suivantes :
   - Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
   - Pour passer à un VPC autre celui par défaut, sous VPC, choisissez un VPC existant, puis sous Subnets (Sous-réseaux), sélectionnez chaque sous-réseau.
7. Pour ajouter des instances Amazon EC2 à votre cluster, développez Infrastructure, puis sélectionnez Instances Amazon EC2. Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :
   - Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :
     - Pour Fonctionnement system/Architecture, choisissez Amazon Linux 2.
     - Dans EC2 instance type (Type d'instance EC2), choisissez `t2.micro`.

       Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
     - Dans le champ Capacité, entrez 1.
8. Cliquez sur Créer.

Étape 2 : créer un groupe de sécurité pour des instances Amazon EC2 et le système de fichiers Amazon EFS

Lors de cette étape, vous créez un groupe de sécurité pour vos instances Amazon EC2 autorisant le trafic réseau entrant sur le port 80 et pour votre système de fichiers Amazon EFS autorisant l'accès entrant depuis vos instances de conteneur.

Créez un groupe de sécurité pour vos instances Amazon EC2 avec les options suivantes :

- Nom du groupe de sécurité : un nom unique pour votre groupe de sécurité.
- VPC : le VPC identifié précédemment pour votre cluster.
- Règle entrante
  - Type : HTTP.
  - Source - 0,0,0.0/0.

Créez un groupe de sécurité pour votre système de fichiers Amazon EFS avec les options suivantes :

- Nom du groupe de sécurité : un nom unique pour votre groupe de sécurité. Par exemple : `EFS-access-for-sg-`*`dc025fa2`*.
- VPC : le VPC identifié précédemment pour votre cluster.
- Règle entrante
  - Type : NFS.
  - Source : personnalisée avec l'ID du groupe de sécurité que vous avez créé pour vos instances.

Pour plus d'informations sur la création d'un groupe de sécurité, consultez la section Création d'un groupe de sécurité pour votre instance Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Étape 3 : Créer un système de fichiers Amazon EFS

Dans cette étape, vous créez un système de fichiers Amazon EFS.

Pour créer un système de fichiers Amazon EFS pour des tâches Amazon ECS.

1. Ouvrez la console Amazon Elastic File System à l'adresse https://console.aws.amazon.com/efs/.
2. Choisissez Create file system (Créer un système de fichiers).
3. Saisissez un nom pour votre système de fichiers, puis choisissez le VPC dans lequel vos instances de conteneur sont hébergées. Par défaut, chaque sous-réseau du VPC spécifié reçoit

une cible de montage qui utilise le groupe de sécurité par défaut de ce VPC. Choisissez ensuite Personnaliser.

> **Note**
>
> Ce didacticiel part du principe que votre système de fichiers Amazon EFS, votre cluster Amazon ECS, vos instances de conteneur et vos tâches se trouvent dans le même VPC. Pour plus d'informations sur le montage d'un système de fichiers à partir d'un autre VPC, consultez la section Procédure pas à pas : montage d'un système de fichiers à partir d'un autre VPC dans le Guide de l'utilisateur Amazon EFS.

4. Sur la page Paramètres du système de fichiers, configurez les paramètres facultatifs, puis sous Paramètres de performance, choisissez le mode de débit Transmission en rafales pour votre système de fichiers. Après avoir configuré les paramètres, sélectionnez Suivant.

   a. (Facultatif) Ajoutez des étiquettes pour votre système de fichiers. Par exemple, vous pouvez spécifier un nom unique pour le système de fichiers en entrant ce nom dans la colonne Valeur en regard de la clé Nom.

   b. (Facultatif) Activez la gestion du cycle de vie pour économiser de l'argent sur un stockage rarement utilisé. Pour en savoir plus, consultez la section Gestion du cycle de vie EFS du Amazon Elastic File System User Guide.

   c. (Facultatif) Activez le chiffrement. Cochez la case pour activer le chiffrement de votre système de fichiers Amazon EFS au repos.

5. Sur la page Accès réseau, sous Monter les cibles, remplacez la configuration de groupe de sécurité existante de chaque zone de disponibilité par le groupe de sécurité que vous avez créé pour le système de fichiers dans Étape 2 : créer un groupe de sécurité pour des instances Amazon EC2 et le système de fichiers Amazon EFS, puis choisissez Suivant.

6. Il n'est pas nécessaire de configurer la Stratégie de système de fichiers pour ce didacticiel. Vous pouvez donc ignorer la section en choisissant Suivant.

7. Vérifiez les options de votre système de fichiers et choisissez Créer pour terminer le processus.

8. Sur l'écran Systèmes de fichiers, enregistrez l'ID du système de fichiers. À l'étape suivante, vous référencerez cette valeur dans votre définition de tâche Amazon ECS.

Étape 4 : Ajouter du contenu au système de fichiers Amazon EFS

Au cours de cette étape, vous monterez le système de fichiers Amazon EFS sur une instance Amazon EC2 et y ajouterez du contenu. Cette opération servira à tester la nature persistante des données dans ce didacticiel. Lorsque vous utilisez cette fonction, vous avez normalement votre application ou une autre méthode d'écriture de données dans votre système de fichiers Amazon EFS.

Pour créer une instance Amazon EC2 et monter le système de fichiers Amazon EFS

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Choisissez Launch Instances (Lancer les instances).
3. Sous Images d'applications et de systèmes d'exploitation (Amazon Machine Image), sélectionnez l'AMI Amazon Linux 2 (HVM).
4. Sous Type d'instance, conservez le type d'instance par défaut `t2.micro`.
5. Sous Paire de clés (connexion), sélectionnez une paire de clés pour l'accès SSH à l'instance.
6. Sous Paramètres réseau, sélectionnez le VPC que vous avez spécifié pour votre système de fichiers Amazon EFS et votre cluster Amazon ECS. Sélectionnez un sous-réseau et le groupe de sécurité d'instance créé dans Étape 2 : créer un groupe de sécurité pour des instances Amazon EC2 et le système de fichiers Amazon EFS. Configurez le groupe de sécurité de l'instance. Assurez-vous que Auto-assign IP publique est activé.
7. Sous Configurer le stockage, cliquez sur le bouton Modifier pour les systèmes de fichiers, puis choisissez EFS. Sélectionnez le système de fichiers créé dans Étape 3 : Créer un système de fichiers Amazon EFS. Vous pouvez éventuellement modifier le point du montage ou conserver la valeur par défaut.

   > **Important**
   >
   > Vous devez sélectionner un sous-réseau avant de pouvoir ajouter un système de fichiers à l'instance.

8. Désactivez Créer et associer automatiquement des groupes de sécurité. Laissez l'autre case cochée. Choisissez Add shared file system (Ajouter un système de fichiers partagé).
9. Sous Advanced Details (Détails avancés), assurez-vous que le script de données utilisateur est renseigné automatiquement avec les étapes de montage du système de fichiers Amazon EFS.
10. Sous Récapitulatif, assurez-vous que le Nombre d'instances est égal à 1. Choisissez Launch instance (Lancer une instance).

11. Sur la page Lancer une instance, choisissez Afficher toutes les instances pour afficher l'état de vos instances. Initialement, le statut d'État de l'instance est `PENDING`. Une fois que l'état est passé à `RUNNING` et que l'instance a réussi toutes les vérifications de statut, celle-ci est prête à être utilisée.

Maintenant, connectez-vous à l'instance Amazon EC2 et ajoutez du contenu au système de fichiers Amazon EFS.

Pour vous connecter à l'instance Amazon EC2 et ajouter du contenu au système de fichiers Amazon EFS

1. Envoyez une requête SSH à l'instance Amazon EC2 que vous avez créée. Pour plus d'informations, consultez la section Connexion à votre instance Linux à l'aide de SSH dans le Guide de l'utilisateur Amazon EC2.
2. Dans la fenêtre du terminal, exécutez la commande df -T pour vérifier que le système de fichiers Amazon EFS est bien monté. Dans la sortie suivante, nous avons mis en évidence le montage du système de fichiers Amazon EFS.

```
$ df -T
Filesystem     Type            1K-blocks    Used        Available Use% Mounted on
devtmpfs       devtmpfs           485468       0           485468   0% /dev
tmpfs          tmpfs              503480       0           503480   0% /dev/shm
tmpfs          tmpfs              503480     424           503056   1% /run
tmpfs          tmpfs              503480       0           503480   0% /sys/fs/
cgroup
/dev/xvda1     xfs               8376300 1310952          7065348  16% /
127.0.0.1:/    nfs4     9007199254739968       0 9007199254739968   0% /mnt/efs/fs1
tmpfs          tmpfs              100700       0           100700   0% /run/
user/1000
```

3. Accédez au répertoire dans lequel le système de fichiers Amazon EFS est monté. Dans l'exemple précédent, c'est `/mnt/efs/fs1`.
4. Créez un fichier nommé `index.html` avec le contenu suivant:

```
<html>
    <body>
        <h1>It Works!</h1>
        <p>You are using an Amazon EFS file system for persistent container
 storage.</p>
    </body>
```

```
</html>
```

Étape 5 : Créer une définition de tâche

La définition de tâche suivante crée un volume de données nommé `efs-html`. Le conteneur `nginx` monte le volume de données de l'hôte à la racine NGINX, `/usr/share/nginx/html`.

Pour créer une nouvelle définition de tâche à l'aide de la console Amazon ECS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, copiez et collez le texte JSON suivant, en remplaçant `fileSystemId` par l'ID de votre système de fichiers Amazon EFS.

```
{
    "containerDefinitions": [
        {
            "memory": 128,
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "mountPoints": [
                {
                    "containerPath": "/usr/share/nginx/html",
                    "sourceVolume": "efs-html"
                }
            ],
            "name": "nginx",
            "image": "public.ecr.aws/docker/library/nginx:latest"
        }
    ],
    "volumes": [
        {
            "name": "efs-html",
```

```
            "efsVolumeConfiguration": {
                "fileSystemId": "fs-1324abcd",
                "transitEncryption": "ENABLED"
            }
        }
    ],
    "family": "efs-tutorial",
    "executionRoleArn":"arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

> **Note**
>
> Le rôle IAM d'exécution des tâches Amazon ECS ne nécessite aucun Amazon spécifique EFS-related autorisations pour monter un système de fichiers Amazon EFS. Par défaut, s'il n'existe aucune politique basée sur les ressources Amazon EFS, l'accès est accordé à tous les principaux (*) lors de la création du système de fichiers.
> Le rôle de tâche Amazon ECS n'est requis que si l'option « Autorisation IAM EFS » est activée dans la définition de tâche Amazon ECS. Lorsque cette option est activée, l'identité du rôle de la tâche doit être autorisée à accéder au système de fichiers Amazon EFS conformément à la politique basée sur les ressources Amazon EFS, et l'accès anonyme doit être désactivé.

5. Cliquez sur Créer.

Étape 6 : Exécuter une tâche et afficher les résultats

Maintenant que votre système de fichiers Amazon EFS est créé et qu'il existe du contenu web pour le conteneur NGINX à diffuser, vous pouvez exécuter une tâche à l'aide de la définition de tâche que vous avez créée. Le serveur web NGINX diffuse votre page HTML simple. Si vous mettez à jour le contenu de votre système de fichiers Amazon EFS, ces changements sont propagés vers tous les conteneurs qui ont également monté ce système de fichiers.

La tâche s'exécute dans le sous-réseau que vous avez défini pour le cluster.

Pour exécuter une tâche et afficher les résultats à l'aide de la console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster dans lequel la tâche autonome doit s'exécuter.

Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. b. Sous l'onglet Tasks (Tâches), choisissez Run new task (Exécuter une nouvelle tâche).
Type de lancement	a. Sur la page Task (Tâche), choisissez la définition de tâche. b. S'il y a plusieurs révisions, sélectionnez-en une. c. Choisissez Create (Créer), Exécuter la tâche (Exécuter la tâche).

3. (Facultatif) Choisissez comment votre tâche planifiée est distribuée dans votre infrastructure de cluster. Développer Compute configuration (Configuration de calcul), puis procédez comme suit :

Méthode de distribution	Étapes
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch Type (Type de lancement), choisissez EC2.

4. Pour Application type (Type d'application), choisissez Task (Tâche).
5. Pour Task Definition (Définition de tâche), choisissez la définition de tâche `efs-tutorial` que vous avez créée précédemment.
6. Pour Desired tasks (Tâches souhaitées), saisissez 1.
7. Cliquez sur Créer.
8. Sur la page Cluster, choisissez Infrastructure.
9. Sous Instances de conteneur, sélectionnez l'instance de conteneur à laquelle vous connecter.
10. Sur la page Container Instance (Instance de conteneur), sous Networking (Mise en réseau), enregistrez Public IP (IP publique) pour votre instance.
11. Ouvrez un navigateur et saisissez l'adresse IP publique. Vous devez voir le message suivant :

```
It works!
You are using an Amazon EFS file system for persistent container storage.
```

> **Note**
>
> Si le message ne s'affiche pas, assurez-vous que le groupe de sécurité de votre instance de conteneur autorise le trafic réseau entrant sur le port 80 et que le groupe de sécurité de votre système de fichiers autorise l'accès entrant depuis l'instance de conteneur.

Utilisation de FSx for Windows File Server avec Amazon ECS

FSx for Windows File Server fournit des serveurs de fichiers Windows entièrement gérés, qui s'appuient sur un système de fichiers Windows. Lorsque vous utilisez FSx for Windows File Server avec ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d'informations, veuillez consulter Qu'est-ce qu'Amazon FSx for Windows File Server ?.

> **Note**
>
> Instances EC2 qui utilisent Amazon ECS-Optimized L'AMI complète de Windows Server 2016 ne prend pas en charge les volumes de tâches FSx pour Windows File Server ECS.

Vous ne pouvez pas utiliser les volumes FSx for Windows File Server dans un conteneur Windows sur une configuration Fargate. Au lieu de cela, vous pouvez modifier les conteneurs pour les monter au démarrage.

Vous pouvez utiliser FSx for Windows File Server pour déployer des applications Windows nécessitant un accès au stockage externe partagé, au stockage régional hautement disponible ou au stockage haut débit. Vous pouvez monter un ou plusieurs volumes du système de fichiers Amazon FSx for Windows File Server sur un conteneur Amazon ECS qui s'exécute sur une instance Amazon ECS Windows. Vous pouvez partager les volumes du système de fichiers Amazon FSx for Windows File Server entre plusieurs conteneurs Amazon ECS au sein d'une seule tâche Amazon ECS.

Pour activer l'utilisation de FSx for Windows File Server avec ECS, vous devez inclure l'identifiant de système de fichiers FSx for Windows File Server et les informations connexes dans une définition de tâche. Il s'agit de l'extrait de code JSON de définition de tâche suivant. Avant de créer et d'exécuter une définition de tâche, vous avez besoin des éléments suivants.

- Une instance ECS Windows EC2 jointe à un domaine valide. Elle peut être hébergée par AWS Directory Service for Microsoft Active Directory, Active Directory sur site ou Active Directory autohébergé sur Amazon EC2.
- Un AWS Secrets Manager paramètre secret ou Systems Manager qui contient les informations d'identification utilisées pour rejoindre le domaine Active Directory et joindre le système de fichiers FSx pour Windows File Server. Les valeurs d'informations d'identification sont les informations d'identification de nom et de mot de passe que vous avez entrées lors de la création de l'Active Directory.

Pour voir un didacticiel connexe, consultez Configuration des systèmes de fichiers FSx for Windows File Server pour Amazon ECS.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez des volumes FSx for Windows File Server :

- Les volumes FSx pour Windows File Server sont pris en charge de manière native avec Amazon ECS sur les instances Windows Amazon EC2. Amazon ECS gère automatiquement le montage via la configuration par définition des tâches.

Sur les instances Amazon EC2 Linux, Amazon ECS ne peut pas monter automatiquement FSx pour les volumes du serveur de fichiers Windows via des définitions de tâches. Cependant, vous pouvez monter manuellement un partage de fichiers FSx pour Windows File Server sur une instance Linux EC2 au niveau de l'hôte, puis monter ce chemin par liaison dans vos conteneurs Amazon ECS. Pour plus d'informations, voir Montage de partages de fichiers Amazon FSx depuis Linux.

> **Important**
>
> Il s'agit d'une configuration autogérée. Pour obtenir des conseils sur le montage et la maintenance des partages de fichiers FSx pour Windows File Server sous Linux, consultez Documentation relative au serveur de fichiers FSx pour Windows.

> **Important**
>
> Lorsque vous utilisez un partage de serveur de fichiers FSx pour Windows monté manuellement sur des instances Linux EC2, Amazon ECS et FSx pour Windows File Server fonctionnent indépendamment : Amazon ECS ne surveille pas le montage Amazon FSx et FSx pour Windows File Server ne suit pas le placement des tâches Amazon ECS ni les événements du cycle de vie. Vous êtes chargé de garantir l'accessibilité du réseau entre vos instances de conteneur Amazon ECS et le système de fichiers Amazon FSx, de mettre en œuvre des contrôles de l'état des montages et de gérer la logique de reconnexion pour tolérer les événements de basculement.

- FSx for Windows File Server avec Amazon ECS ne prend pas en charge AWS Fargate.
- FSx for Windows File Server avec Amazon ECS n’est pas pris en charge sur les instances gérées Amazon ECS.
- FSx for Windows File Server avec Amazon ECS avec le mode réseau `awsvpc` nécessite la version `1.54.0` ou une version ultérieure de l'agent de conteneur.
- Le nombre maximal de lettres de lecteur pouvant être utilisées pour une tâche Amazon ECS est de 23. Chaque tâche avec un volume FSx for Windows File Server reçoit une lettre de lecteur qui lui est attribuée.
- Par défaut, le temps de nettoyage des ressources de tâches est de trois heures après la fin de la tâche. Même si aucune tâche ne l'utilise, un mappage de fichier créé par une tâche persiste

pendant trois heures. Vous pouvez configurer la durée de nettoyage par défaut à l'aide de la variable d'environnement Amazon ECS. `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

- En général, les tâches s'exécutent uniquement sur le même VPC que le système de fichiers FSx for Windows File Server. Toutefois, il est possible de prendre en charge plusieurs VPC s'il existe une connectivité réseau établie entre le VPC de cluster Amazon ECS et le système de fichiers FSx for Windows File Server via l'appairage de VPC.
- Vous contrôlez l'accès à un système de fichiers FSx for Windows File Server au niveau du réseau en configurant les groupes de sécurité VPC. Seules les tâches hébergées sur des instances EC2 jointes au domaine Active Directory avec des groupes de sécurité Active Directory correctement configurés peuvent accéder au partage de fichiers FSx for Windows File Server. Si les groupes de sécurité sont mal configurés, Amazon ECS ne parvient pas à lancer la tâche et affiche le message d'erreur suivant : « `unable to mount file system` *`fs-id`*. »
- FSx pour Windows File Server est intégré à Gestion des identités et des accès AWS (IAM) pour contrôler les actions que vos utilisateurs et groupes IAM peuvent effectuer sur des ressources spécifiques de FSx pour Windows File Server. Avec l'autorisation client, les clients peuvent définir des rôles IAM qui autorisent ou refusent l'accès à des systèmes de fichiers FSx for Windows File Server spécifiques, éventuellement exiger un accès en lecture seule et éventuellement autoriser ou interdire l'accès racine au système de fichiers à partir du client. Pour plus d'informations, veuillez consulter la rubrique Sécurité dans le Guide de l'utilisateur Amazon FSx Windows.

Pratiques exemplaires en matière d'utilisation de FSx for Windows File Server avec Amazon ECS

Tenez compte des recommandations suivantes en matière de pratique exemplaire lorsque vous utilisez FSx for Windows File Server avec Amazon ECS.

Sécurité et contrôles d'accès pour FSx for Windows File Server

FSx for Windows File Server offre les fonctionnalités de contrôle d'accès suivantes que vous pouvez utiliser pour garantir que les données stockées dans un système de fichiers FSx for Windows File Server sont sécurisées et accessibles uniquement à partir des applications qui en ont besoin.

Chiffrement des données pour les volumes FSx for Windows File Server

FSx for Windows File Server prend en charge deux formes de chiffrement pour les systèmes de fichiers. Il s'agit du chiffrement des données en transit et du chiffrement au repos. Le chiffrement des données en transit est pris en charge sur les partages de fichiers mappés sur une instance de conteneur compatible avec le protocole SMB 3.0 ou une version ultérieure. Le chiffrement des

données au repos est automatiquement activé lors de la création d'un système de fichiers Amazon FSx. Amazon FSx utilise le chiffrement SMB pour chiffrer automatiquement les données en transit lorsque vous accédez à votre système de fichiers, sans que vous ayez à modifier vos applications. Pour plus d'informations, consultez la section Chiffrement des données dans Amazon FSx dans le Guide de l'utilisateur Amazon FSx for Windows File Server.

Utilisation des ACL Windows pour le contrôle d'accès au niveau des dossiers

L'instance Windows Amazon EC2 accède aux partages de fichiers Amazon FSx à l'aide des informations d'identification Active Directory. Elle utilise des listes de contrôle d'accès (ACL) Windows standard pour un contrôle d'accès précis au niveau des fichiers et des dossiers. Vous pouvez créer plusieurs informations d'identification, chacune correspondant à un dossier spécifique au sein du partage qui est mappé à une tâche spécifique.

Dans l'exemple suivant, la tâche a accès au dossier `App01` à l'aide d'informations d'identification enregistrées dans Secrets Manager. Son Amazon Resource Name (ARN) est `1234`.

```
"rootDirectory": "\\path\\to\\my\\data\App01",
"credentialsParameter": "arn-1234",
"domain": "corp.fullyqualified.com",
```

Dans un autre exemple, une tâche a accès au dossier `App02` à l'aide d'informations d'identification enregistrées dans Secrets Manager. Son ARN est `6789`.

```
"rootDirectory": "\\path\\to\\my\\data\App02",
"credentialsParameter": "arn-6789",
"domain": "corp.fullyqualified.com",
```

Spécification d'un système de fichiers FSx for Windows File Server dans une définition de tâche Amazon ECS

Pour utiliser des volumes de système de fichiers FSx for Windows File Server pour vos conteneurs, spécifiez les configurations de volume et de point de montage dans votre définition de tâche. L'extrait de JSON de définition de tâche indiqué ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
```

```
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType file
 -Value '<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>It Works!</h2> <p>You are using Amazon
 FSx for Windows File Server file system for persistent container storage.</p>' -
Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force; Start-
Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -Destination C:\
\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
```

```
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

`FSxWindowsFileServerVolumeConfiguration`

Type : objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez le système de fichiers FSx for Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers FSx for Windows File Server à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : oui

Répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l'intérieur de l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : chaîne

Obligatoire : oui

Les options d'informations d'identification d'autorisation :

- Amazon Resource Name (ARN) pour un secret Secrets Manager.
- Amazon Resource Name (ARN) pour un paramètre Systems Manager.

`domain`

Type : chaîne

Obligatoire : oui

Un nom de domaine complet hébergé par un répertoire AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou un Active Directory EC2 autohébergé.

Méthodes de stockage des informations d'identification du volume FSx for Windows File Server

Il existe deux méthodes différentes de stockage des informations d'identification à utiliser avec le paramètre d'informations d'identification.

- AWS Secrets Manager secret

  Ces informations d'identification peuvent être créées dans le AWS Secrets Manager console à l'aide de Autre type de secret catégorie. Vous ajoutez une ligne pour chaque key/value paire, username/admin et mot de passe/*`password`*.
- Paramètre Systems Manager

Ces informations d'identification peuvent être créées dans la console de paramètres Systems Manager en saisissant du texte dans le formulaire qui se trouve dans l'exemple d'extrait de code suivant.

```
{
  "username": "admin",
  "password": "password"
}
```

Le paramètre `credentialsParameter` dans la définition de tâche `FSxWindowsFileServerVolumeConfiguration` contient l'ARN de secret ou l'ARN du paramètre Systems Manager. Pour de plus amples informations, veuillez consulter les rubriques Présentation d' AWS Secrets Manager dans le Guide de l'utilisateur de Secrets Manager et Systems Manager Parameter Store dans le Guide de l'utilisateur de Systems Manager.

Configuration des systèmes de fichiers FSx for Windows File Server pour Amazon ECS

Découvrez comment lancer un Amazon ECS-Optimized Instance Windows qui héberge un système de fichiers FSx pour Windows File Server et des conteneurs pouvant accéder au système de fichiers. Pour ce faire, vous devez d'abord créer un Directory Service AWS Microsoft Active Directory géré. Créez ensuite un système de fichiers FSx for Windows File Server et un cluster avec une instance Amazon EC2 et une définition de tâche. Vous configurez la définition de tâche pour vos conteneurs afin qu'ils utilisent le système de fichiers FSx for Windows File Server. Enfin, vous testez le système de fichiers.

Il faut 20 à 45 minutes chaque fois que vous lancez ou supprimez le système de fichiers Active Directory ou FSx for Windows File Server. Préparez-vous à réserver au moins 90 minutes pour terminer le didacticiel ou le compléter en quelques sessions.

Prérequis pour le didacticiel

- Un utilisateur administratif. Consultez Configurer l'utilisation d'Amazon ECS.
- (Facultatif) Une paire de clés `PEM` permettant de vous connecter à votre instance Windows EC2 via un accès RDP. Pour plus d'informations sur la création de paires de clés, consultez les sections Paires de clés Amazon EC2 et Instances Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.
- Un VPC avec au moins un sous-réseau public et un sous-réseau privé et un groupe de sécurité. Vous pouvez utiliser votre VPC par défaut. Vous n'avez pas besoin d'une passerelle ou d'un

périphérique NAT. Directory Service ne prend pas en charge la traduction d'adresses réseau (NAT) avec Active Directory. Pour que cela fonctionne, le répertoire Active Directory, le système de fichiers FSx for Windows File Server, le cluster ECS et l'instance EC2 doivent se trouver dans votre VPC. Pour plus d'informations sur les VPC et Active Directories, consultez Création d'un VPC et Conditions préalables à la création d'un AWS Microsoft AD géré.

- Les écs de l'IAMInstanceRole et EcsTaskExecutionRole les autorisations sont associées à votre compte. Ces rôles liés à un service permettent aux services d'effectuer des appels d'API et d'accéder aux conteneurs, secrets, répertoires et serveurs de fichiers en votre nom.

Étape 1 : Créer des rôles IAM d'accès

Créez un cluster avec Console de gestion AWS.

1. Voir Rôle IAM d'instance de conteneur Amazon ECS pour vérifier si vous avez un ECSInstanceRole et pour voir comment vous pouvez en créer un si vous n'en avez pas.
2. Nous recommandons que les stratégies de rôle soient personnalisées pour les autorisations minimales dans un environnement de production réel. Dans le cadre de ce didacticiel, vérifiez que les points suivants AWS une politique gérée est attachée à votre ECSInstanceRole. Joignez la stratégie si elle n'est pas déjà attachée.

   - AmazonEC2ContainerServiceforEC2Role
   - AmazonSSMManagedInstanceCore
   - AmazonSSMDirectoryServiceAccess

   À joindre AWS politiques gérées.

   a. Ouvrez la console IAM.
   b. Dans le panneau de navigation, choisissez Rôles.
   c. Choisissez un rôle géré par AWS .
   d. Choisissez Autorisations, Attacher des stratégies.
   e. Pour réduire le nombre de stratégies disponibles à attacher, utilisez Filter (Filtrer).
   f. Sélectionnez la stratégie appropriée, puis choisissez Attach policy (Attacher la stratégie).
3. Voir Rôle IAM d'exécution de tâche Amazon ECS pour vérifier si vous avez un ECSTaskExecutionRole et pour voir comment vous pouvez en créer un si vous n'en avez pas.

Nous recommandons que les stratégies de rôle soient personnalisées pour les autorisations minimales dans un environnement de production réel. Dans le cadre de ce didacticiel, vérifiez que les points suivants AWS les politiques gérées sont rattachées à votre ECSTaskExecutionRole. Attachez les stratégies si elles ne sont pas déjà attachées. Utilisez la procédure indiquée dans la section précédente pour joindre le AWS politiques gérées.

- SecretsManagerReadWrite
- AmazonFSxReadOnlyAccess
- AmazonSSMReadOnlyAccess
- AmazonECSTaskExecutionRolePolicy

Étape 2 : Créer un Active Directory (AD) Windows

1. Suivez les étapes décrites dans Création de votre AWS Microsoft AD géré dans le AWS Guide d'administration du service d'annuaire. Utilisez le VPC que vous avez désigné pour ce didacticiel. À l'étape 3 de Création de votre AWS Microsoft AD géré, enregistrez le nom d'utilisateur et le mot de passe administrateur pour les utiliser lors de l'étape suivante. Notez également le nom DNS complet du répertoire pour les prochaines étapes. Vous pouvez réaliser l'étape suivante pendant la création d'Active Directory.
2. Créez un AWS Secret du gestionnaire de secrets à utiliser dans les étapes suivantes. Pour plus d'informations, voir Commencez avec Secrets Manager dans le AWS Guide de l'utilisateur de Secrets Manager.

   a. Ouvrez la console Secrets Manager.

   b. Cliquez sur Store a new secret (Stocker un nouveau secret).

   c. Sélectionnez Other type of secrets (Autres types de secrets).

   d. Pour Secret key/value, dans la première ligne, créez une clé **`username`** avec valeur **`admin`**. Cliquez sur + Add row (+ Ajouter une ligne).

   e. Dans la nouvelle ligne, créez une clé **`password`**. Dans le champ valeur, saisissez le mot de passe que vous avez saisi à l'étape 3 de Créez votre AWS Annuaire AD géré.

   f. Cliquez sur Next (Suivant).

   g. Fournissez un nom et une description de secret. Cliquez sur Suivant.

   h. Cliquez sur Suivant. Cliquez sur Store (Stocker).

   i. À partir de la page Secrets, cliquez sur le secret que vous venez de créer.

j. Enregistrez l'ARN du nouveau secret pour l'utiliser dans les étapes suivantes.

k. Vous pouvez passer à l'étape suivante pendant la création de votre répertoire Active Directory.

Étape 3 : Vérifier et mettre à jour votre groupe de sécurité

Dans cette étape, vous vérifiez et mettez à jour les règles du groupe de sécurité que vous utilisez. Pour cela, vous pouvez utiliser le groupe de sécurité par défaut qui a été créé pour votre VPC.

Vérifiez et mettez à jour le groupe de sécurité.

Vous devez créer ou modifier votre groupe de sécurité pour envoyer des données depuis et vers les ports. Celles-ci sont décrites dans la section Groupes de sécurité Amazon VPC du Guide de l'utilisateur FSx for Windows File Server. Pour ce faire, créez la règle entrante du groupe de sécurité affichée dans la première ligne du tableau suivant des règles entrantes. Cette règle autorise le trafic entrant à partir d'interfaces réseau (et de leurs instances associées) affectées au groupe de sécurité. Toutes les ressources cloud que vous créez se trouvent au sein du même VPC et sont rattachées au même groupe de sécurité. Par conséquent, cette règle autorise l'envoi du trafic vers et depuis le système de fichiers FSx for Windows File Server, Active Directory et l'instance ECS selon les besoins. Les autres règles entrantes autorisent le trafic à servir le site web et l'accès RDP pour la connexion à votre instance ECS.

Le tableau suivant indique les règles entrantes de groupe de sécurité requises pour ce didacticiel.

Type	Protocole	Plage de ports	Source
Tout le trafic	Tous	Tous	*sg-securitygroup*
HTTPS	TCP	443	0,0,0.0/0
RDP	TCP	3389	l'adresse IP de votre ordinateur portable

Le tableau suivant indique les règles sortantes du groupe de sécurité requises pour ce didacticiel.

Type	Protocole	Plage de ports	Destination
Tout le trafic	Tous	Tous	0,0,0.0/0

1. Ouvrez la console EC2 et sélectionnez Security groups (Groupes de sécurité) depuis le menu de gauche.
2. Dans la liste des groupes de sécurité qui s'affiche, cochez la case située à gauche du groupe de sécurité que vous utilisez pour ce didacticiel.

   Les informations de votre groupe de sécurité s'affichent.
3. Modifiez les règles entrantes et sortantes en sélectionnant l'onglet Inbound rules (Règles entrantes) ou Outbound rules (Règles sortantes) et en cliquant sur Edit inbound rules (Modifier les règles entrantes) ou Edit outbound rules (Modifier les règles sortantes). Modifiez les règles pour qu'elles correspondent à celles affichées dans les tableaux précédents. Lorsque vous aurez créé votre instance EC2 plus tard dans ce didacticiel, modifiez la règle entrante RDP source avec l’adresse IP publique de votre instance EC2, comme décrit dans la section Connexion à votre instance Windows à l’aide du protocole RDP du Guide de l’utilisateur Amazon EC2.

Étape 4 :Créer un système de fichiers Amazon FSx for Windows File Server

Une fois que votre groupe de sécurité est vérifié et mis à jour et que votre répertoire Active Directory est créé avec l'état active (actif), créez le système de fichiers FSx for Windows File Server dans le même VPC que votre répertoire Active Directory. Suivez les étapes suivantes pour créer un système de fichiers FSx for Windows File Server.

Créez votre premier système de fichiers.

1. Ouvrez la console Amazon FSx.
2. Dans Sur le tableau de bord, choisissez Create file system (Créer un système de fichiers) pour ouvrir l'assistant de création de système de fichiers.
3. Sur la page Select file system type (Sélectionner le type de système de fichiers), choisissez FSx for Windows File Server, puis Next (Suivant). La page Create file system (Créer un système de fichiers) s'affiche.
4. Dans la section File system details (Informations du système de fichiers), indiquez un nom pour votre système de fichiers. Donner un nom à vos systèmes de fichiers facilite leur recherche et

leur gestion. Vous pouvez utiliser jusqu'à 256 caractères Unicode. Les caractères autorisés sont les lettres, les chiffres, les espaces et les caractères spéciaux suivants : signe plus (+), signe moins (-), signe égal (=), point (.), trait de soulignement (_), deux points (:) et barre oblique (/).

5. Pour Type de déploiement choisir Single-AZ pour déployer un système de fichiers déployé dans une seule zone de disponibilité. Single-AZ 2 est la dernière génération de systèmes de fichiers à zone de disponibilité unique et prend en charge le stockage sur SSD et sur disque dur.
6. Pour Storage type (Type de stockage), choisissez HDD.
7. Pour Storage capacity (Capacité de stockage), saisissez la capacité de stockage minimale.
8. Conservez la valeur par défaut de Throughput capacity (Capacité de débit).
9. Dans le Réseau et sécurité section, choisissez le même Amazon VPC que vous avez choisi pour votre Directory Service annuaire.
10. Pour VPC Security Groups (Groupes de sécurité VPC), choisissez le groupe de sécurité que vous avez vérifié à l'Étape 3 : Vérifier et mettre à jour votre groupe de sécurité.
11. Pour Windows authentication (Authentification Windows), choisissez AWS Managed Microsoft Active Directory (Annuaire Microsoft Active Directory géré par), puis choisissez votre répertoire Directory Service dans la liste.
12. Pour Chiffrement, conservez la valeur par défaut Clé de chiffrement réglage de aws/fsx (par défaut).
13. Conservez les paramètres par défaut pour Maintenance preferences (Préférences de maintenance).
14. Cliquez sur Next (Suivant).
15. Vérifiez la configuration du système de fichiers qui s'affiche sur la page Create file system (Créer un système de fichiers). Pour référence, notez les paramètres du système de fichiers que vous pouvez modifier une fois le système de fichiers créé. Choisissez Create file system (Créer un système de fichiers).
16. Notez l'ID du système de fichiers. Vous en aurez besoin dans une étape ultérieure.

    Vous pouvez passer aux étapes suivantes pour créer un cluster et une instance EC2 pendant la création du système de fichiers FSx for Windows File Server.

Étape 5 : créer un cluster Amazon ECS

Création d'un cluster à l'aide de la console Amazon ECS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, pour Nom du cluster, saisissez windows-fsx-cluster.
6. Élargir Infrastructures, clair AWS Fargate (sans serveur), puis sélectionnez Instances Amazon EC2.

    - Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :

        - Pour Fonctionnement system/Architecture, choisissez Windows Server 2019 Core.
        - Pour Type d'instance EC2, choisissez t2.medium ou t2.micro.
7. Cliquez sur Créer.

Étape 6 : créer une instance Amazon EC2 optimisée pour Amazon ECS

Créez une instance de conteneur Windows Amazon ECS.

Pour créer une instance Amazon ECS

1. Utilisez la commande `aws ssm get-parameters` pour récupérer le nom d'AMI de la région qui héberge votre VPC. Pour plus d'informations, voir Récupération d'Amazon ECS-Optimized Métadonnées AMI.
2. Utilisez la console Amazon EC2 pour lancer l'instance.

    a. Ouvrez la console Amazon EC2 à l’adresse https://console.aws.amazon.com/ec2/.

    b. Dans la barre de navigation, sélectionnez la région à utiliser.

    c. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une instance).

    d. Pour Name (Nom), saisissez un nom unique.

    e. Pour Images de l'application et du SE (Amazon Machine Image), dans le champ de recherche, saisissez le nom de l'AMI que vous avez récupérée.

    f. Pour Type d'instance, choisissez t2.medium ou t2.micro.

    g. Pour Key pair (login) (Paire de clés (connexion)), choisissez une paire de clés. Si vous ne spécifiez pas de paire de clés, vous

h. Sous Paramètres réseau, pour VPC et Sous-réseau, choisissez votre VPC et un sous-réseau public.

i. Sous Network Settings (Paramètres réseau), pour Security group (Groupe de sécurité), choisissez un groupe de sécurité existant ou créez-en un nouveau. Assurez-vous que le groupe de sécurité que vous choisissez possède les règles entrantes et sortantes définies dans Prérequis pour le didacticiel.

j. En dessous Paramètres réseau, pour Auto-assign IP publique, sélectionnez Activer.

k. Développez Détails avancés, puis pour Répertoire de jonction de domaines, sélectionnez l'ID de l'Active Directory que vous avez créé. Cette option de jonction de domaines joint votre répertoire AD lorsque l'instance EC2 est lancée.

l. En dessous Détails avancés, pour Profil d'instance IAM , choisissez ecsInstanceRole.

m. Configurez votre instance de conteneur Amazon ECS avec les données utilisateur suivantes. En dessous Détails avancés, collez le script suivant dans le Données de l'utilisateur champ, remplacement *cluster_name* avec le nom de votre cluster.

```
<powershell>
Initialize-ECSAgent -Cluster windows-fsx-cluster -EnableTaskIAMRole
</powershell>
```

n. Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).

o. Une page de confirmation indique que l'instance est en cours de lancement. Sélectionnez View Instances (Afficher les instances) pour fermer la page de confirmation et revenir à la console.

3. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

4. Dans le panneau de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.

5. Choisissez l'onglet Infrastructure et vérifiez que votre instance a été enregistrée dans le cluster windows-fsx-cluster.

Étape 7 : Enregistrement d'une définition de tâche Windows

Avant de pouvoir exécuter des conteneurs Windows dans votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. L'exemple de définition de tâche suivant affiche une page Web simple. La tâche lance deux conteneurs qui ont accès au système de fichiers FSx. Le premier

conteneur écrit un fichier HTML dans le système de fichiers. Le deuxième conteneur télécharge le fichier HTML à partir du système de fichiers et sert la page web.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, remplacez les valeurs pour votre rôle d'exécution de tâche et les détails sur votre système de fichiers FSx, puis choisissez Enregistrer.

```
{
    "containerDefinitions": [
        {
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "portMappings": [],
            "command": ["New-Item -Path C:\\fsx-windows-dir\\index.html -ItemType
 file -Value '<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>It
 Works!</h2> <p>You are using Amazon FSx for Windows File Server file system for
 persistent container storage.</p>' -Force"],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": false,
            "name": "container1",
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ]
        },
        {
            "entryPoint": [
                "powershell",
                "-Command"
```

```
            ],
            "portMappings": [
                {
                    "hostPort": 443,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "command": ["Remove-Item -Recurse C:\\inetpub\\wwwroot\\* -Force;
 Start-Sleep -Seconds 120; Move-Item -Path C:\\fsx-windows-dir\\index.html -
Destination C:\\inetpub\\wwwroot\\index.html -Force; C:\\ServiceMonitor.exe
 w3svc"],
            "mountPoints": [
                {
                    "sourceVolume": "fsx-windows-dir",
                    "containerPath": "C:\\fsx-windows-dir",
                    "readOnly": false
                }
            ],
            "cpu": 512,
            "memory": 256,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "essential": true,
            "name": "container2"
        }
    ],
    "family": "fsx-windows",
    "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
    "volumes": [
        {
            "name": "fsx-windows-dir",
            "fsxWindowsFileServerVolumeConfiguration": {
                "fileSystemId": "fs-0eeb5730b2EXAMPLE",
                "authorizationConfig": {
                    "domain": "example.com",
                    "credentialsParameter": "arn:arn-1234"
                },
                "rootDirectory": "share"
            }
        }
    ]
}
```

Étape 8 : Exécuter une tâche et afficher les résultats

Avant d'exécuter la tâche, vérifiez que l'état de votre système de fichiers FSx for Windows File Server est Available (Disponible). Si c'est le cas, vous pouvez exécuter une tâche à l'aide de la définition de tâche que vous avez créée. La tâche commence par créer des conteneurs qui remanient un fichier HTML sur eux à l'aide du système de fichiers. Après le remaniement, un serveur web sert la page HTML simple.

> **Note**
>
> Il se peut que vous ne parveniez pas à vous connecter au site web depuis un VPN.

Exécutez une tâche et affichez les résultats à l'aide de la console Amazon ECS.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.
3. Choisissez l'onglet Tâches, puis Exécuter une nouvelle tâche.
4. Pour Launch Type (Type de lancement), choisissez EC2.
5. Sous Configuration du déploiement, pour Définition de tâche, sélectionnez fsx-windows, puis sélectionnez Créer.
6. Lorsque le statut de votre tâche est EN COURS D'EXÉCUTION, sélectionnez l'ID de la tâche.
7. Sous Conteneurs, lorsque le statut container1 est ARRÊTÉ, sélectionnez container2 pour afficher les détails du conteneur.
8. Dans Informations sur le conteneur pour container2, sélectionnez Liaisons réseau, puis cliquez sur l'adresse IP externe associée au conteneur. Votre navigateur s'ouvre et affiche le message suivant.

```
Amazon ECS Sample App
It Works!
You are using Amazon FSx for Windows File Server file system for persistent
 container storage.
```

> **Note**
>
> L'affichage du message peut prendre quelques minutes. Si ce message ne s'affiche pas après plusieurs minutes, vérifiez que vous n'exécutez pas dans un VPN et assurez-vous

que le groupe de sécurité de votre instance de conteneur autorise le trafic HTTP réseau entrant sur le port 443.

Étape 9 : nettoyer

> **Note**
>
> La suppression du système de fichiers FSx for Windows File Server ou du répertoire AD prend 20 à 45 minutes. Vous devez attendre que les opérations de suppression du système de fichiers FSx for Windows File Server soient terminées avant de lancer les opérations de suppression du répertoire AD.

Supprimez le système de fichiers FSx for Windows File Server.

1. Ouvrez la console Amazon FSx.
2. Cliquez sur la case d'option située à gauche du système de fichiers FSx for Windows File Server que vous venez de créer.
3. Choisissez Actions.
4. Sélectionnez Delete file system (Supprimer le système de fichiers).

Supprimez le répertoire AD.

1. Ouvrez la console Directory Service.
2. Cliquez sur la case d'option située à gauche du répertoire AD que vous venez de créer.
3. Choisissez Actions.
4. Sélectionnez Delete directory (Supprimer le répertoire).

Supprimez le cluster.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters, puis choisissez windows-fsx-cluster.
3. Choisissez Supprimer le cluster.
4. Saisissez l'expression, puis choisissez Supprimer.

Résiliez l'instance EC2.

1. Ouvrez la console Amazon EC2.
2. Dans le menu de gauche, sélectionnez Instances.
3. Cochez la case située à gauche de l'instance EC2 que vous avez créée.
4. Cliquez sur État de l'instance, puis sur Résilier l'instance.

Supprimez le secret.

1. Ouvrez la console Secrets Manager.
2. Sélectionnez le secret que vous avez créé pour cette démonstration.
3. Cliquez sur Actions.
4. Sélectionnez Delete secret (Supprimer le secret).

Configuration des fichiers S3 pour Amazon ECS

S3 Files est un système de fichiers partagé qui connecte n'importe quel AWS ressources de calcul directement avec vos données dans Amazon S3. Il fournit un accès rapide et direct à toutes vos données S3 sous forme de fichiers avec une sémantique complète du système de fichiers et des performances à faible latence, sans que vos données ne quittent jamais S3. Vous pouvez lire, écrire et organiser les données à l'aide d'opérations sur les fichiers et les répertoires, tandis que S3 Files synchronise automatiquement votre système de fichiers et votre compartiment S3. Avec Amazon ECS, vous pouvez définir les systèmes de fichiers S3 comme des volumes dans vos définitions de tâches, ce qui permet à vos conteneurs d'accéder directement au système de fichiers aux données stockées dans des compartiments S3. Pour en savoir plus sur Amazon S3 Files et ses fonctionnalités, consultez le Guide de l'utilisateur d'Amazon S3.

Disponibilité

La prise en charge des fichiers S3 dans Amazon ECS est disponible pour les types de lancement suivants lors de la disponibilité générale :

- Fargate — Entièrement pris en charge.
- Instances gérées Amazon ECS — Entièrement pris en charge.
- Amazon EC2 — Entièrement pris en charge.

Considérations

- Le système de fichiers S3 utilise un `s3filesVolumeConfiguration` paramètre dans la définition de la tâche.
- Le système de fichiers S3 nécessite un Amazon Resource Name (ARN) complet pour identifier le système de fichiers. Le format ARN est le suivant :

```
arn:{partition}:s3files:{region}:{account-id}:file-system/fs-xxxxx
```

- Le chiffrement en transit est obligatoire pour les volumes du système de fichiers S3 et est automatiquement appliqué. Il n'existe aucune option pour le désactiver.
- Le rôle IAM de la tâche est obligatoire pour les volumes du système de fichiers S3 et est automatiquement appliqué. Il n'existe aucune option pour le désactiver.

Conditions préalables

Avant de configurer les volumes du système de fichiers S3 dans vos définitions de tâches Amazon ECS, assurez-vous que les conditions préalables suivantes sont remplies :

- Un système de fichiers S3 et une cible de montage — Un système de fichiers S3 doit être créé et associé à un compartiment S3. Pour obtenir des instructions sur la création d'un système de fichiers S3, consultez Guide de l'utilisateur d'Amazon S3 Files.
- Un rôle IAM dans une tâche — Votre définition de tâche doit inclure un rôle IAM de tâche doté des autorisations suivantes :
  - Autorisations de connexion et d'interaction avec les systèmes de fichiers S3 à partir du code de votre application (exécutée dans le conteneur).
  - Autorisations pour lire les objets S3 à partir du code de votre application (exécutée dans le conteneur).
- Configuration du VPC et du groupe de sécurité — Votre système de fichiers S3 doit être accessible depuis le VPC et les sous-réseaux sur lesquels s'exécutent vos tâches Amazon ECS.
- Points d'accès aux fichiers S3 (Facultatif) — Si vous souhaitez appliquer des contrôles d'accès spécifiques à l'application, créez un point d'accès S3 Files et indiquez l'ARN dans la définition de la tâche.

Pour plus d'informations, reportez-vous à prérequis pour les fichiers S3.

Spécifiez un volume Amazon S3 Files dans votre définition de tâche Amazon ECS

Vous pouvez configurer les volumes S3 Files dans vos définitions de tâches Amazon ECS à l'aide de la console Amazon ECS, AWS CLI, ou le AWS API.

Utilisation de la console Amazon ECS

1. Ouvrez la console Amazon ECS à l'adresse https://console.aws.amazon.com/ecs/.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Créer une nouvelle définition de tâche ou sélectionnez une définition de tâche existante et créez une nouvelle révision.
4. Dans le Infrastructures section, assurez-vous que vous disposez d'un rôle Task IAM configuré avec les autorisations requises.
5. Dans le Rangement section, choisissez Ajouter du volume.
6. Pour Type de volume, sélectionnez Fichiers S3.
7. Pour ARN du système de fichiers, entrez l'ARN complet de votre système de fichiers S3. Le format ARN est le suivant :

   ```
   arn:{partition}:s3files:{region}:{account-id}:file-system/fs-xxxxx
   ```

8. (Facultatif) Pour Répertoire racine, entrez le chemin du système de fichiers à monter en tant que racine. Si elle n'est pas spécifiée, la racine du système de fichiers (`/`) est utilisé.
9. (Facultatif) Pour Port de chiffrement de transit, entrez le numéro de port pour l'envoi de données chiffrées entre l'hôte Amazon ECS et le système de fichiers S3. Si vous ne spécifiez pas de port de chiffrement en transit, il utilise la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS.
10. (Facultatif) Pour ARN du point d'accès, sélectionnez le point d'accès S3 Files à utiliser dans la liste déroulante.
11. Dans le Points de montage du conteneur section, sélectionnez le conteneur et spécifiez le chemin de montage local dans votre conteneur où le volume doit être monté à l'intérieur du conteneur.
12. Choisissez Créez pour créer la définition de la tâche.

Utilisation de AWS CLI

Pour spécifier un volume S3 Files dans une définition de tâche à l'aide du AWS CLI, utilisez le `register-task-definition` commande avec `s3filesVolumeConfiguration` paramètre dans la définition du volume.

Voici un exemple d'extrait JSON de définition de tâche qui définit un volume S3 Files et le monte dans un conteneur :

```
{
  "family": "s3files-task-example",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
  "containerDefinitions": [
    {
      "name": "my-container",
      "image": "my-image:latest",
      "essential": true,
      "mountPoints": [
        {
          "containerPath": "/mnt/s3data",
          "sourceVolume": "my-s3files-volume"
        }
      ]
    }
  ],
  "volumes": [
    {
      "name": "my-s3files-volume",
      "s3filesVolumeConfiguration": {
        "fileSystemArn": "arn:aws:s3files:us-east-1:123456789012:file-
system/fs-0123456789abcdef0",
        "rootDirectory": "/",
        "transitEncryptionPort": 2999
      }
    }
  ]
}
```

Enregistrez la définition de tâche :

```
aws ecs register-task-definition --cli-input-json file://s3files-task-def.json
```

Pour utiliser un point d'accès, incluez le `accessPointArn` paramètre :

```
{
  "name": "my-s3files-volume",
  "s3filesVolumeConfiguration": {
    "fileSystemArn": "arn:aws:s3files:us-east-1:123456789012:file-
system/fs-0123456789abcdef0",
    "rootDirectory": "/",
    "transitEncryptionPort": 2999,
    "accessPointArn": "arn:aws:s3files:us-east-1:123456789012:file-
system/fs-0123456789abcdef0/access-point/fsap-0123456789abcdef0"
  }
}
```

Paramètres de configuration du volume des fichiers S3

Le tableau suivant décrit les paramètres disponibles dans `s3filesVolumeConfiguration` objet :

`fileSystemArn`

Type : chaîne

Obligatoire : oui

L'ARN complet du système de fichiers S3 à monter. Format : `arn:{partition}:s3files:{region}:{account-id}:file-system/fs-xxxxx`

`rootDirectory`

Type : chaîne

Obligatoire : non

Le répertoire du système de fichiers S3 à monter en tant que racine du volume. La valeur par défaut est `/` si elle n'est pas précisée.

`transitEncryptionPort`

Type : Integer

Obligatoire : non

Port à utiliser pour envoyer des données chiffrées entre l'hôte Amazon ECS et le système de fichiers S3. Le chiffrement du transit lui-même est toujours activé et ne peut pas être désactivé.

`accessPointArn`

Type : chaîne

Obligatoire : non

L'ARN complet du point d'accès S3 Files à utiliser. Les points d'accès fournissent des points d'entrée spécifiques à l'application dans le système de fichiers avec des paramètres d'identité utilisateur et de répertoire racine imposés.

Utilisation de volumes Docker avec Amazon ECS

Lorsque vous utilisez des volumes Docker, le pilote `local` intégré ou un pilote de volume tiers peut être utilisé. Les volumes Docker sont gérés par Docker et un répertoire est créé dans `/var/lib/docker/volumes` sur l'instance de conteneur qui contient les données du volume.

Pour utiliser des volumes Docker, spécifiez un `dockerVolumeConfiguration` dans votre définition de tâche. Pour plus d’informations, consultez la section Volumes dans la documentation Docker.

Certains cas d'utilisation courants pour les volumes Docker sont les suivants :

- Fournir des volumes de données permanent à utiliser avec des conteneurs
- Partager un volume de données défini à différents emplacements sur différents conteneurs situés sur la même instance de conteneur
- Définir un volume de données vide, non permanent et le monter dans plusieurs conteneurs au sein d’une même tâche
- Fournir un volume de données à votre tâche qui est gérée par un pilote tiers

Considérations relatives à l’utilisation des volumes Docker

Tenez compte des éléments suivants lorsque vous utilisez des volumes Docker :

- Les volumes Docker sont pris en charge uniquement en utilisant le type de lancement EC2 ou des instances externes.
- Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`.
- Si un pilote tiers est utilisé, assurez-vous qu'il est installé et actif sur l'instance de conteneur avant le démarrage de l'agent de conteneur. Si le pilote tiers n'est pas actif avant le démarrage de l'agent, vous pouvez redémarrer l'agent de conteneur à l'aide de l'une des commandes suivantes :

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2 :

```
sudo systemctl restart ecs
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux :

```
sudo stop ecs && sudo start ecs
```

Pour plus d’informations sur la manière de spécifier un volume Docker dans une définition de tâche, consultez la section Spécification d’un volume Docker dans une définition de tâche Amazon ECS.

Spécification d’un volume Docker dans une définition de tâche Amazon ECS

Avant que vos conteneurs puissent utiliser des volumes de données, vous devez spécifier les configurations du volume et du point de montage dans votre définition de tâche. Cette section décrit la configuration du volume pour un conteneur. Pour les tâches qui utilisent un volume Docker, spécifiez une propriété `dockerVolumeConfiguration`. Pour les tâches qui utilisent un volume hôte de montage lié, spécifiez un `host` et éventuellement un `sourcePath`.

La définition de tâche JSON indiquée ci-dessous illustre la syntaxe des objets `volumes` et `mountPoints` pour un conteneur :

```
{
    "containerDefinitions": [
        {
            "mountPoints": [
                {
                    "sourceVolume": "string",
                    "containerPath": "/path/to/mount_volume",
                    "readOnly": boolean
                }
            ]
        }
    ],
    "volumes": [
        {
            "name": "string",
            "dockerVolumeConfiguration": {
                "scope": "string",
                "autoprovision": boolean,
                "driver": "string",
```

```
                "driverOpts": {
                    "key": "value"
                },
                "labels": {
                    "key": "value"
                }
            }
        }
    ]
}
```

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union (`-`) et traits de soulignement (`_`) sont autorisés. Ce nom est référencé dans le paramètre `sourceVolume` de l'objet `mountPoints` de définition du conteneur.

`dockerVolumeConfiguration`

Tapez : DockerVolumeConfiguration Objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker sont pris en charge uniquement lors de l'exécution de tâches sur des instances EC2. Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits

lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : Boolean

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n’est utilisé que si `scope` a la valeur `shared`. Si le champ `scope` a la valeur `task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker, car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l’aide de l’interface CLI du plug-in Docker, utilisez `docker plugin ls` pour récupérer le nom du pilote à partir de votre instance de conteneur. Si le pilote a été installé à l’aide d’une autre méthode, utilisez la découverte du plug-in Docker pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Mappage des options spécifiques au pilote Docker à transmettre. Ce paramètre correspond à `DriverOpts` dans la section « Créer un volume » de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur des instances EC2 exécutant le système d'exploitation Windows, laissez la valeur `false` par défaut.

Exemples de volumes Docker pour Amazon ECS

Les exemples suivants montrent comment fournir un stockage éphémère pour un conteneur et comment fournir un volume partagé pour plusieurs conteneurs, et comment fournir un stockage persistant NFS pour un conteneur.

Pour fournir un stockage éphémère à un conteneur à l'aide d'un volume Docker

Dans cet exemple, un conteneur utilise un volume de données vide qui est éliminé une fois la tâche terminée. Un exemple d'utilisation est que vous pourriez avoir un conteneur qui doit accéder à un emplacement de stockage de fichiers de travail pendant une tâche. Cette tâche peut être réalisée à l'aide d'un volume Docker.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, nous spécifions la portée sous la forme `task`. Le volume est donc supprimé après l'arrêt de la tâche et il utilise le pilote `local` intégré.

```
"volumes": [
    {
        "name": "scratch",
        "dockerVolumeConfiguration" : {
            "scope": "task",
            "driver": "local",
            "labels": {
                "scratch": "space"
            }
        }
    }
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

```
"containerDefinitions": [
    {
        "name": "container-1",
        "mountPoints": [
            {
              "sourceVolume": "scratch",
```

```
                "containerPath": "/var/scratch"
            }
        ]
    }
]
```

Pour fournir un stockage persistant à plusieurs conteneurs à l'aide d'un volume Docker

Dans cet exemple, vous voulez un volume partagé qui sera utilisé par plusieurs conteneurs et vous souhaitez qu'il persiste après l'arrêt des tâches qui l'utilisent. Le pilote `local` intégré est en cours d'utilisation. Ainsi, le volume est toujours lié au cycle de vie de l'instance de conteneur.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, spécifiez une portée `shared` pour que le volume persiste, définissez autoprovision sur `true`. C'est ainsi que le volume est créé pour être utilisé. Ensuite, utilisez également le pilote `local` intégré.

   ```
   "volumes": [
       {
           "name": "database",
           "dockerVolumeConfiguration" : {
               "scope": "shared",
               "autoprovision": true,
               "driver": "local",
               "labels": {
                   "database": "database_name"
               }
           }
       }
   ]
   ```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

   ```
   "containerDefinitions": [
       {
           "name": "container-1",
           "mountPoints": [
           {
             "sourceVolume": "database",
   ```

```
          "containerPath": "/var/database"
        }
      ]
    },
    {
      "name": "container-2",
      "mountPoints": [
        {
          "sourceVolume": "database",
          "containerPath": "/var/database"
        }
      ]
    }
  ]
```

Pour fournir un stockage permanent NFS pour un conteneur à l'aide d'un volume Docker

Dans cet exemple, un conteneur utilise un volume de données NFS qui est automatiquement monté lorsque la tâche démarre et démonté lorsque la tâche s'arrête. Cela utilise le pilote `local` intégré Docker. Un exemple d'utilisation est que vous pourriez avoir un stockage NFS local et que vous devez y accéder à l'aide d'une tâche ECS Anywhere. Cela peut être réalisé à l'aide d'un volume Docker avec option de pilote NFS.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données avec les valeurs `name` et `DockerVolumeConfiguration`. Dans cet exemple, spécifiez une portée `task` de telle sorte que le volume soit démonté une fois la tâche terminée. Utilisez le pilote `local` et configurez le `driverOpts` avec le `type`, `device` et les options o en conséquence. Remplacez `NFS_SERVER` par le point de terminaison du serveur NFS.

```
"volumes": [
       {
           "name": "NFS",
           "dockerVolumeConfiguration" : {
               "scope": "task",
               "driver": "local",
               "driverOpts": {
                   "type": "nfs",
                   "device": "$NFS_SERVER:/mnt/nfs",
                   "o": "addr=$NFS_SERVER"
               }
           }
```

```
        }
    ]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec des valeurs `mountPoints` qui font référence au nom du volume défini et la valeur `containerPath` sur laquelle monter le volume sur le conteneur.

```
"containerDefinitions": [
        {
            "name": "container-1",
            "mountPoints": [
                {
                  "sourceVolume": "NFS",
                  "containerPath": "/var/nfsmount"
                }
            ]
        }
    ]
```

Utilisation de montages liés avec Amazon ECS

Avec les montages liés, un fichier ou un répertoire sur un hôte, tel qu'une instance Amazon EC2, est monté dans un conteneur. Les montages liés sont pris en charge lorsque vous exécutez des tâches sur des instances Fargate ou Amazon EC2. Les montages liés sont liés au cycle de vie du conteneur qui les utilise. Une fois que tous les conteneurs qui utilisent un montage lié sont arrêtés, par exemple lorsqu'une tâche est arrêtée, les données sont supprimées. Pour les tâches hébergées sur des instances Amazon EC2, les données peuvent être liées au cycle de vie de l'instance Amazon EC2 hôte en spécifiant un `host` et une valeur `sourcePath` facultative dans votre définition de tâche. Pour de plus amples informations, consultez la section Montages liés dans la documentation Docker.

Voici quelques cas d'utilisation courants pour les montages liés.

- Pour fournir un volume de données vide à monter dans un ou plusieurs conteneurs.
- Pour monter un volume de données hôte dans un ou plusieurs conteneurs.
- Pour partager un volume de données d'un conteneur source avec d'autres conteneurs dans la même tâche.
- Pour exposer un chemin d'accès et son contenu d'un fichier Dockerfile à un ou plusieurs conteneurs.

Considérations relatives à l'utilisation des montages liés

Lorsque vous utilisez des montages liés, tenez compte des éléments suivants.

- Par défaut, les tâches hébergées sur AWS Fargate en utilisant la version de la plateforme `1.4.0` ou version ultérieure (Linux) ou `1.0.0` ou version ultérieure (Windows) reçoivent un minimum de 20 Go de stockage éphémère pour les montages par liaison. Vous pouvez augmenter la quantité totale de stockage éphémère jusqu'à un maximum de 200 Gio en spécifiant le paramètre `ephemeralStorage` dans votre définition de tâche.
- Pour exposer des fichiers d'un fichier Dockerfile à un volume de données lorsqu'une tâche est exécutée, le plan de données Amazon ECS recherche une directive `VOLUME`. Si le chemin absolu spécifié dans la directive `VOLUME` est le même que le `containerPath` spécifié dans la définition de tâche, les données du chemin de directive `VOLUME` sont copiées sur le volume de données. Dans l'exemple Dockerfile suivant, un fichier nommé `examplefile` dans le répertoire `/var/log/exported` est écrit sur l'hôte, puis monté à l'intérieur du conteneur.

  ```
  FROM public.ecr.aws/amazonlinux/amazonlinux:latest
  RUN mkdir -p /var/log/exported
  RUN touch /var/log/exported/examplefile
  VOLUME ["/var/log/exported"]
  ```

  Par défaut, les autorisations de volume sont définies sur `0755` et le propriétaire en tant que `root`. Vous pouvez personnaliser ces autorisations dans le fichier Dockerfile. L'exemple suivant définit le propriétaire du répertoire en tant que `node`.

  ```
  FROM public.ecr.aws/amazonlinux/amazonlinux:latest
  RUN yum install -y shadow-utils && yum clean all
  RUN useradd node
  RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
  RUN touch /var/log/exported/examplefile
  USER node
  VOLUME ["/var/log/exported"]
  ```

- Pour les tâches hébergées sur des instances Amazon EC2, lorsque les valeurs `host` et `sourcePath` ne sont pas spécifiées, le démon Docker gère le montage lié à votre place. Lorsqu'aucun conteneur ne fait référence à ce montage lié, le service de nettoyage des tâches de l'agent de conteneur Amazon ECS finit par le supprimer. Par défaut, cela se produit trois heures après la sortie du conteneur. Toutefois, vous pouvez configurer cette durée avec la variable de l'agent `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`. Pour plus d'informations, consultez

Configuration de l'agent de conteneur Amazon ECS. Si vous avez besoin de conserver ces données au-delà du cycle de vie de conteneur, spécifiez une valeur `sourcePath` pour le montage lié.

- Pour les tâches hébergées sur des instances gérées Amazon ECS, certaines parties du système de fichiers racine sont en lecture seule. Read/write les montages bind doivent utiliser des répertoires inscriptibles tels que `/var` pour les données persistantes ou `/tmp` pour les données temporaires. Tentative de création read/write lier les montages à d'autres répertoires entraîne l'échec du lancement de la tâche avec une erreur similaire à la suivante :

```
error creating empty volume: error while creating volume path '/path': mkdir /path:
 read-only file system
```

Read-only supports de liaison (configurés avec `"readOnly": true` dans le `mountPoints` paramètre) peut pointer vers n'importe quel répertoire accessible sur l'hôte.

Pour afficher la liste complète des chemins accessibles en écriture, vous pouvez exécuter une tâche sur une instance gérée Amazon ECS et l'utiliser pour inspecter la table de montage de l'instance. Créez une définition de tâche avec les paramètres suivants pour accéder au système de fichiers hôte :

```
{
    "pidMode": "host",
    "containerDefinitions": [{
        "privileged": true,
        ...
    }]
}
```

Exécutez ensuite les commandes suivantes depuis le conteneur :

```
# List writable mounts
cat /proc/1/root/proc/1/mounts | awk '$4 ~ /^rw,/ || $4 == "rw" {print $2}' | sort

# List read-only mounts
cat /proc/1/root/proc/1/mounts | awk '$4 ~ /^ro,/ || $4 == "ro" {print $2}' | sort
```

> **Important**
>
> Le `privileged` le paramètre accorde au conteneur des fonctionnalités étendues sur l'hôte, équivalentes à un accès root. Dans cet exemple, il est utilisé pour inspecter la table de montage de l'hôte à des fins de diagnostic. Pour plus d'informations, consultez Évitez d'exécuter des conteneurs dotés de privilèges (Amazon EC2).

Pour plus d'informations sur l'exécution interactive de commandes dans des conteneurs, consultez Surveillance des conteneurs Amazon ECS avec ECS Exec.

Spécification d'un montage lié dans une définition de tâche Amazon ECS

Pour les tâches Amazon ECS hébergées sur des instances Fargate ou Amazon EC2, l'extrait JSON de définition de tâche suivant montre la syntaxe des objets `volumes`, `mountPoints` et `ephemeralStorage` pour une définition de tâche.

```
{
   "family": "",
   ...
   "containerDefinitions" : [
      {
         "mountPoints" : [
            {
               "containerPath" : "/path/to/mount_volume",
               "sourceVolume" : "string"
            }
          ],
          "name" : "string"
       }
    ],
    ...
    "volumes" : [
       {
          "name" : "string"
       }
    ],
    "ephemeralStorage": {
    "sizeInGiB": integer
    }
```

```
}
```

Pour les tâches Amazon ECS hébergées sur des instances Amazon EC2, vous pouvez utiliser le paramètre `host` facultatif et un `sourcePath` lorsque vous spécifiez les détails du volume de tâches. Lorsqu'il est spécifié, il lie le montage lié au cycle de vie de la tâche plutôt qu'au conteneur.

```
"volumes" : [
    {
        "host" : {
            "sourcePath" : "string"
        },
        "name" : "string"
    }
]
```

Voici des descriptions plus détaillées de chaque paramètre de définition de tâche.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union (`-`) et traits de soulignement (`_`) sont autorisés. Ce nom est référencé dans le paramètre `sourceVolume` de l'objet `mountPoints` de définition du conteneur.

`host`

Obligatoire : non

Le paramètre `host` est utilisé pour lier le cycle de vie du montage lié à l'instance Amazon EC2 hôte, plutôt qu'à la tâche et à l'endroit où elle est stockée. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` Le paramètre est pris en charge uniquement lors de l'utilisation de tâches hébergées sur des instances Amazon EC2 ou des instances gérées Amazon ECS.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le paramètre `host` est utilisé, spécifiez un paramètre `sourcePath` pour déclarer le chemin d'accès sur l'instance Amazon EC2 hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le paramètre `host` contient un emplacement de fichier `sourcePath`, le volume de données persiste à l'emplacement spécifié sur l'instance Amazon EC2 hôte jusqu'à ce que vous le supprimiez manuellement. Si la valeur `sourcePath` n'existe pas sur l'instance Amazon EC2 hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur des instances EC2 exécutant le système d'exploitation Windows, laissez la valeur `false` par défaut.

`ephemeralStorage`

Type : objet

Obligatoire : non

Quantité de magasin éphémère à allouer pour la tâche. Ce paramètre est utilisé pour étendre la quantité totale de stockage éphémère disponible, au-delà de la quantité par défaut, pour les tâches hébergées sur AWS Fargate en utilisant la version de la plateforme `1.4.0` ou version ultérieure (Linux) ou `1.0.0` ou version ultérieure (Windows).

Vous pouvez utiliser la CLI Copilot, CloudFormation, le AWS SDK ou interface de ligne de commande pour spécifier le stockage éphémère pour un montage par liaison.

Exemples de montage lié pour Amazon ECS

Les exemples suivants couvrent les cas d'utilisation courants pour l'utilisation d'un montage lié pour vos conteneurs.

Pour allouer une quantité accrue d'espace de stockage éphémère pour une tâche Fargate

Pour les tâches Amazon ECS hébergées sur Fargate à l'aide de la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows), vous pouvez allouer plus que la quantité de

magasins éphémères par défaut pour les conteneurs de votre tâche à utiliser. Cet exemple peut être intégré dans les autres exemples afin d'allouer plus de stockage éphémère pour vos tâches Fargate.

- Dans la définition de la tâche, définissez un objet `ephemeralStorage`. La valeur `sizeInGiB` doit être un nombre entier compris entre `21` et `200` est exprimée en GiB.

```
"ephemeralStorage": {
    "sizeInGiB": integer
}
```

Pour fournir un volume de données vide pour un ou plusieurs conteneurs

Dans certains cas, vous voudrez fournir aux conteneurs dans une tâche un peu d'espace vide. Supposons par exemple que deux conteneurs de base de données doivent accéder au même emplacement de stockage temporaire pendant une tâche. Cela peut être réalisé à l'aide d'un montage lié.

1. Dans la section `volumes` de la définition de tâche, définissez un montage lié nommé `database_scratch`.

```
  "volumes": [
    {
      "name": "database_scratch"
    }
  ]
```

2. Dans la section `containerDefinitions`, créez les définitions de conteneur de base de données. Cela permet de monter le volume.

```
"containerDefinitions": [
    {
      "name": "database1",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
```

```
        }
      ]
    },
    {
      "name": "database2",
      "image": "my-repo/database",
      "cpu": 100,
      "memory": 100,
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "database_scratch",
          "containerPath": "/var/scratch"
        }
      ]
    }
  ]
```

Pour exposer un chemin d'accès et son contenu d'un fichier Dockerfile à un ou plusieurs conteneurs.

Dans cet exemple, vous avez un fichier Dockerfile qui écrit les données que vous souhaitez monter à l'intérieur d'un conteneur. Cet exemple fonctionne pour les tâches hébergées sur des instances Fargate ou Amazon EC2.

1. Créez un fichier Dockerfile. L'exemple suivant utilise l'image du conteneur Amazon Linux 2 publique et crée un fichier nommé `examplefile` dans le répertoire `/var/log/exported` que nous voulons monter à l'intérieur du conteneur. Le répertoire `VOLUME` doit spécifier un chemin absolu.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN mkdir -p /var/log/exported
   RUN touch /var/log/exported/examplefile
   VOLUME ["/var/log/exported"]
   ```

   Par défaut, les autorisations de volume sont définies sur `0755` et le propriétaire en tant que `root`. Ces autorisations peuvent être modifiées dans le fichier Dockerfile. L'exemple suivant définit le propriétaire du répertoire `/var/log/exported` en tant que `node`.

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   RUN yum install -y shadow-utils && yum clean all
   ```

```
RUN useradd node
RUN mkdir -p /var/log/exported && chown node:node /var/log/exported
USER node
RUN touch /var/log/exported/examplefile
VOLUME ["/var/log/exported"]
```

2. Dans la section `volumes` de la définition de tâche, définissez un volume nommé `application_logs`.

```
"volumes": [
  {
    "name": "application_logs"
  }
]
```

3. Dans la section `containerDefinitions`, créez les définitions de conteneur d'application. Cela permet de monter le stockage. La valeur `containerPath` doit correspondre au chemin absolu spécifié dans la directive `VOLUME` du Dockerfile.

```
"containerDefinitions": [
  {
    "name": "application1",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
      }
    ]
  },
  {
    "name": "application2",
    "image": "my-repo/application",
    "cpu": 100,
    "memory": 100,
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "application_logs",
        "containerPath": "/var/log/exported"
```

```
        }
      ]
    }
]
```

Pour fournir un volume de données vide pour un conteneur lié au cycle de vie de l'instance Amazon EC2 hôte

Pour les tâches hébergées sur des instances Amazon EC2, vous pouvez utiliser des montages liés et lier les données au cycle de vie de l'instance Amazon EC2 hôte. Vous pouvez effectuer cette opération en utilisant le paramètre `host` et en spécifiant une valeur `sourcePath`. Tous les fichiers qui existent sur le `sourcePath` sont présentés dans les conteneurs à la valeur `containerPath`. Tous les fichiers écrits dans la valeur `containerPath` sont écrits dans la valeur `sourcePath` de l'instance Amazon EC2 hôte.

> **Important**
>
> Amazon ECS ne synchronise pas votre stockage entre les instances Amazon EC2. Les tâches qui utilisent un stockage permanent peuvent être placées sur n'importe quelle instance Amazon EC2 disposant de la capacité nécessaire dans votre cluster. Si vos tâches nécessitent un stockage permanent après l'arrêt et le redémarrage, spécifiez toujours la même instance Amazon EC2 au moment du lancement de la tâche avec le AWS CLI tâche de démarrage commande. Vous pouvez également utiliser des volumes Amazon EFS pour le stockage permanent. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.

1. Dans la section `volumes` de la définition de tâche, définissez un montage lié avec les valeurs `name` et `sourcePath`. Dans l'exemple suivant, l'instance Amazon EC2 hôte contient des données sous `/ecs/webdata` que vous souhaitez monter à l'intérieur du conteneur.

```
"volumes": [
  {
    "name": "webdata",
    "host": {
      "sourcePath": "/ecs/webdata"
    }
  }
```

```
]
```

2. Dans la section `containerDefinitions`, définissez un conteneur avec une valeur `mountPoints` qui fait référence au nom du montage lié défini et la valeur `containerPath` sur laquelle monter le montage lié sur le conteneur.

```
"containerDefinitions": [
  {
    "name": "web",
    "image": "public.ecr.aws/docker/library/nginx:latest",
    "cpu": 99,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webdata",
        "containerPath": "/usr/share/nginx/html"
      }
    ]
  }
]
```

Pour monter un volume défini sur plusieurs conteneurs à différents emplacements

Vous pouvez définir un volume de données dans une définition de tâche et monter ce volume à différents emplacements de différents conteneurs. Par exemple, votre conteneur hôte possède un dossier de données de site web à l'adresse `/data/webroot`. Vous voudrez peut-être monter ce volume de données en lecture seule sur deux serveurs web qui ont des racines de documents différentes.

1. Dans la section `volumes` de la définition de tâche, définissez un volume de données nommé `webroot` et ayant le chemin source `/data/webroot`.

```
"volumes": [
  {
```

```
      "name": "webroot",
      "host": {
        "sourcePath": "/data/webroot"
      }
    }
  ]
```

2. Dans la section `containerDefinitions`, définissez un conteneur pour chaque serveur web avec des valeurs `mountPoints` qui associent le volume `webroot` à la valeur `containerPath` pointant vers la racine du document pour ce conteneur.

```
"containerDefinitions": [
  {
    "name": "web-server-1",
    "image": "my-repo/ubuntu-apache",
    "cpu": 100,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 80,
        "hostPort": 80
      }
    ],
    "essential": true,
    "mountPoints": [
      {
        "sourceVolume": "webroot",
        "containerPath": "/var/www/html",
        "readOnly": true
      }
    ]
  },
  {
    "name": "web-server-2",
    "image": "my-repo/sles11-apache",
    "cpu": 100,
    "memory": 100,
    "portMappings": [
      {
        "containerPort": 8080,
        "hostPort": 8080
      }
    ],
```

```
      "essential": true,
      "mountPoints": [
        {
          "sourceVolume": "webroot",
          "containerPath": "/srv/www/htdocs",
          "readOnly": true
        }
      ]
    }
  ]
```

Pour monter des volumes depuis un autre conteneur à l'aide **`Volumes de`**

Pour les tâches hébergées sur des instances Amazon EC2, vous pouvez définir un ou plusieurs volumes sur un conteneur, puis utiliser le paramètre `volumesFrom` dans une définition du conteneur différente, au sein de la même tâche, pour monter tous les volumes du `sourceContainer` à leurs points de montage définis à l'origine. Le paramètre `volumesFrom` s'applique aux volumes définis dans la définition de tâche et à ceux qui sont intégrés à l'image au sein d'un Dockerfile.

1. (Facultatif) Pour partager un volume qui est intégré à une image, suivez les instructions `VOLUME` du fichier Dockerfile. L'exemple de Dockerfile suivant utilise une image `httpd`, puis ajoute un volume et le monte à l'emplacement `dockerfile_volume` dans la racine du document Apache. Il s'agit du dossier utilisé par le serveur web `httpd`.

   ```
   FROM httpd
   VOLUME ["/usr/local/apache2/htdocs/dockerfile_volume"]
   ```

   Vous pouvez créer une image avec ce Dockerfile, la transférer dans un référentiel (comme Docker Hub), puis l'utiliser dans votre définition de tâche. L'exemple d'image `my-repo/httpd_dockerfile_volume` utilisée dans les étapes suivantes a été créée avec le Dockerfile précédent.
2. Créez une définition de tâche qui définit les autres volumes et points de montage pour les conteneurs. Dans la section `volumes` de cet exemple, vous créez un volume vide appelé `empty` qui est géré par le démon Docker. Il existe également un volume hôte défini qui est appelé `host_etc`. Il exporte le dossier `/etc` dans l'instance de conteneur hôte.

   ```
   {
     "family": "test-volumes-from",
     "volumes": [
   ```

```
    {
      "name": "empty",
      "host": {}
    },
    {
      "name": "host_etc",
      "host": {
        "sourcePath": "/etc"
      }
    }
  ],
```

Dans la section des définitions de conteneur, créez un conteneur qui monte les volumes définis précédemment. Dans cet exemple, le conteneur `web` monte les volumes `empty` et `host_etc`. Il s'agit du conteneur qui utilise l'image créée avec un volume dans le Dockerfile.

```
"containerDefinitions": [
    {
      "name": "web",
      "image": "my-repo/httpd_dockerfile_volume",
      "cpu": 100,
      "memory": 500,
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "empty",
          "containerPath": "/usr/local/apache2/htdocs/empty_volume"
        },
        {
          "sourceVolume": "host_etc",
          "containerPath": "/usr/local/apache2/htdocs/host_etc"
        }
      ],
      "essential": true
    },
```

Créez un autre conteneur qui utilise `volumesFrom` pour monter tous les volumes qui sont associés au conteneur `web`. Tous les volumes du conteneur `web` sont également montés sur le conteneur `busybox`. Cela inclut le volume spécifié dans le Dockerfile qui a été utilisé pour générer l'image `my-repo/httpd_dockerfile_volume`.

```
    {
      "name": "busybox",
      "image": "busybox",
      "volumesFrom": [
        {
          "sourceContainer": "web"
        }
      ],
      "cpu": 100,
      "memory": 500,
      "entryPoint": [
        "sh",
        "-c"
      ],
      "command": [
        "echo $(date) > /usr/local/apache2/htdocs/empty_volume/date && echo $(date)
 > /usr/local/apache2/htdocs/host_etc/date && echo $(date) > /usr/local/apache2/
htdocs/dockerfile_volume/date"
      ],
      "essential": false
    }
  ]
}
```

Lorsque la tâche est exécutée, les deux conteneurs montent les volumes, et la fonction `command` du conteneur `busybox` écrit la date et l'heure dans un fichier. Ce fichier est appelé `date` dans chacun des dossiers de volumes. Les dossiers sont alors visibles sur le site web affiché par le conteneur `web`.

> **Note**
>
> Étant donné que le conteneur `busybox` exécute une commande rapide avant de s'arrêter, il doit être défini en tant que `"essential": false` dans la définition de conteneur. Dans le cas contraire, il arrête l'ensemble de la tâche lorsqu'il s'arrête.

Gestion de l’espace mémoire d’échange de conteneur sur Amazon ECS

Avec Amazon ECS, vous pouvez contrôler l'utilisation de l'espace mémoire d'échange sur votre Linux-based Instances Amazon EC2 au niveau du conteneur. En utilisant une configuration d'échange par conteneur, l'échange peut être activé ou désactivé pour chaque conteneur au sein d'une définition de tâche. Pour ceux qui l'ont activé, la quantité maximale d'espace d'échange utilisée peut être limitée. Par exemple, l'échange peut être désactivé dans les conteneurs sensibles à la latence. En revanche, les conteneurs avec des demandes de mémoire transitoires élevées peuvent avoir l'échange activé pour réduire les risques d'erreurs de mémoire insuffisante lorsque le conteneur est sous charge.

La configuration d'échange pour un conteneur est gérée par les paramètres de définition de conteneur suivants :

`maxSwap`

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est converti en l’option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur et de la valeur `maxSwap`.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`swappiness`

Vous pouvez utiliser ce paramètre pour régler le comportement d'échange de mémoire d'un conteneur. Une valeur `swappiness` de `0` fait que l'échange ne se produit pas, sauf si nécessaire. Avec la valeur `swappiness` pour `100`, l'échange de pages a lieu de manière agressive. Les valeurs acceptées sont les nombres entiers compris entre `0` et `100`. Si le paramètre `swappiness` n'est pas spécifié, la valeur par défaut `60` est utilisée. Si aucune valeur n'est spécifiée pour `maxSwap`, le paramètre est ignoré. Ce paramètre correspond à l’option `--memory-swappiness` de docker run.

Dans l'exemple suivant, la syntaxe JSON est fournie.

```
"containerDefinitions": [{
        ...
```

```
        "linuxParameters": {
            "maxSwap": integer,
            "swappiness": integer
        },
        ...
}]
```

Considérations

Tenez compte des points suivants lorsque vous utilisez une configuration d'échange par conteneur.

- L'espace d'échange doit être activé et alloué sur l'instance Amazon EC2 hébergeant vos tâches pour que les conteneurs puissent l'utiliser. Par défaut, l'échange n'est pas activé pour les AMI optimisées Amazon ECS. Vous devez activer l'échange sur l'instance pour utiliser cette fonction. Pour plus d'informations, consultez la section Volumes d'échange pour le stockage d'instance dans le Guide de l'utilisateur Amazon EC2 ou Comment allouer de la mémoire en tant qu'espace d'échange dans une instance Amazon EC2 ?.
- Les paramètres de définition de conteneur pour l'espace d'échange ne sont pris en charge que pour les définitions de tâche indiquant EC2. Ces paramètres ne sont pas pris en charge pour les définitions de tâches destinées uniquement à l'utilisation d'Amazon ECS sur Fargate.
- Cette fonction est uniquement prise en charge pour les conteneurs Linux. Les conteneurs Windows ne sont pas pris en charge actuellement.
- Si les paramètres de définition de conteneur `maxSwap` et `swappiness` sont omis d'une définition de tâche, chaque conteneur a une valeur `swappiness` par défaut de `60`. De plus, l'utilisation totale de l'échange est limitée à deux fois la mémoire du conteneur.
- Si vous utilisez des tâches sur Amazon Linux 2023, le paramètre `swappiness` n'est pas pris en charge.

Différences entre les définitions de tâches Amazon ECS pour les instances gérées Amazon ECS

Pour utiliser les instances gérées Amazon ECS, vous devez configurer votre définition de tâche pour utiliser le type de lancement des instances gérées Amazon ECS. Il existe d'autres considérations à prendre en compte lors de l'utilisation d'instances gérées Amazon ECS.

Paramètres de définition de tâche

Les tâches qui utilisent des instances gérées Amazon ECS prennent en charge la plupart des paramètres de définition de tâche Amazon ECS disponibles. Cependant, certains paramètres présentent des comportements ou des limites spécifiques lorsqu'ils sont utilisés avec des tâches d'instances gérées Amazon ECS.

Les paramètres de définition de tâche ne sont pas valides dans les tâches d'instances gérées Amazon ECS :

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `dockerVolumeConfiguration`
- `ephemeralStorage`
- `extraHosts`
- `fsxWindowsFileServerVolumeConfiguration`
- `hostname`
- `inferenceAccelerator`
- `ipcMode`
- `links`
- `maxSwap`
- `proxyConfiguration`
- `sharedMemorySize`
- `sourcepathVolumes`
- `swappiness`
- `tmpfs`

Les paramètres de définition de tâche suivants sont valides dans les tâches d'instances gérées Amazon ECS, mais présentent des limitations à prendre en compte :

- `networkConfiguration` : les tâches d'instances gérées Amazon ECS utilisent le mode réseau `awsvpc` ou `host`.

- `placementConstraints` : les attributs de contrainte suivants sont pris en charge.
  - `ecs.subnet-id`
  - `ecs.availability-zone`
  - `ecs.instance-type`
  - `ecs.cpu-architecture`
- `requiresCompatibilities` : doit inclure `MANAGED_INSTANCES` pour garantir la compatibilité de la définition de tâche avec les instances gérées Amazon ECS.
- `resourceRequirement` : `InferenceAccelerator` n'est pas pris en charge.
- `operatingSystemFamily` : les instances gérées Amazon ECS utilisent `LINUX`.
- `volumes` - Lorsque vous utilisez des supports de fixation avec un `sourcePath`, le chemin doit pointer vers un répertoire accessible en écriture sur l'hôte. Certaines parties du système de fichiers Amazon ECS Managed Instance sont en lecture seule. Les répertoires inscriptibles incluent `/var` et `/tmp`. Pour plus d'informations, consultez Utilisation de montages liés avec Amazon ECS.
- `firelensConfiguration` - Configure le FireLens routeur log. Le `s3` le type de fichier de configuration n'est pas pris en charge ; utilisez `file` type. Pour plus d'informations, consultez Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

Pour vous assurer que votre définition de tâche est valide pour l'utilisation d'instances gérées Amazon ECS, vous pouvez spécifier les éléments suivants lors de l'enregistrement de la définition de tâche :

- Dans le Console de gestion AWS, pour Nécessite des compatibilités champ, spécifiez `MANAGED_INSTANCES`.
- Dans le AWS CLI, spécifiez le `--requires-compatibilities` option.
- Dans l'API Amazon ECS, spécifiez l'indicateur `requiresCompatibilities`.

Différences de définition de tâche Amazon ECS pour Fargate

Pour utiliser Fargate, vous devez configurer votre définition de tâche pour utiliser le type de lancement Fargate. D'autres considérations à prendre en compte lors de l'utilisation de Fargate.

Paramètres de définition de tâche

Les tâches qui utilisent Fargate ne prennent pas en charge tous les paramètres de définition de tâche Amazon ECS disponibles. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment pour les tâches Fargate.

Les paramètres de définition de tâche suivants ne sont pas valides dans les tâches Fargate :

- `disableNetworking`
- `dnsSearchDomains`
- `dnsServers`
- `dockerSecurityOptions`
- `extraHosts`
- `gpu`
- `ipcMode`
- `links`
- `placementConstraints`
- `privileged`
- `maxSwap`
- `swappiness`

Les paramètres de définition de tâche suivants sont valides dans les tâches Fargate, mais avec des limitations à prendre en considération :

- `linuxParameters` — Lors de la spécification Linux-specific options qui sont appliquées au conteneur, pour `capabilities` la seule fonctionnalité que vous pouvez ajouter est `CAP_SYS_PTRACE`. Les paramètres `devices`, `sharedMemorySize` et `tmpfs` ne sont pas pris en charge. Pour plus d'informations, consultez Paramètres Linux.
- `volumes` : les tâches Fargate prennent en charge uniquement les volumes hôtes de montage lié, le paramètre `dockerVolumeConfiguration` n'est donc pas pris en charge. Pour plus d'informations, consultez Volumes.
- `cpu` - Pour les conteneurs Windows sur AWS Fargate, la valeur ne peut pas être inférieure à 1 vCPU.
- `networkConfiguration` : les tâches Fargate utilisent toujours le mode réseau `awsvpc`.

Afin que votre définition de tâche puisse être utilisée avec Fargate, vous pouvez spécifier ce qui suit lors de l'enregistrement de la définition de tâche :

- Dans le Console de gestion AWS, pour Nécessite des compatibilités champ, spécifiez `FARGATE`.
- Dans le AWS CLI, spécifiez le `--requires-compatibilities` option.
- Dans l'API Amazon ECS, spécifiez l'indicateur `requiresCompatibilities`.

Systèmes d’exploitation et architectures

Lorsque vous configurez une tâche et une définition du conteneur pour AWS Fargate, vous devez spécifier le système d'exploitation exécuté par le conteneur. Les systèmes d'exploitation suivants sont pris en charge pour AWS Fargate :

- Amazon Linux 2

> **Note**
>
> Les conteneurs Linux n’utilisent que le noyau et la configuration du noyau du système d’exploitation hôte. Par exemple, la configuration du noyau inclut les commandes du système `sysctl`. Une image de conteneur Linux peut être créée à partir d'une image de base contenant les fichiers et les programmes de n'importe quelle distribution Linux. Si l'architecture du processeur correspond, vous pouvez exécuter des conteneurs à partir de n'importe quelle image de conteneur Linux sur n'importe quel système d'exploitation.

- Windows Server 2019 Full
- Windows Server 2019 Core
- Windows Server 2022 Full
- Windows Server 2022 Core

Lorsque vous exécutez des conteneurs Windows sur AWS Fargate, vous devez disposer de l'architecture de processeur X86_64.

Lorsque vous exécutez des conteneurs Linux sur AWS Fargate, vous pouvez utiliser l'architecture CPU X86_64 ou l'architecture ARM64 pour votre ARM-based candidatures. Pour plus d’informations, consultez the section called “Définitions de tâche pour les charges de travail ARM 64 bits”.

UC et mémoire au niveau de la tâche

Pour les définitions de tâche Amazon ECS pour AWS Fargate, le CPU et la mémoire doivent être spécifiés au niveau de la tâche. La plupart des cas d'utilisation requièrent uniquement la spécification de ces ressources au niveau de la tâche. Le tableau suivant indique les combinaisons valides de CPU et de mémoire au niveau de la tâche. Vous pouvez spécifier des valeurs de mémoire dans la définition de tâche sous forme de chaîne en Mio ou en Go. Par exemple, vous pouvez spécifier une valeur de mémoire de `3072` en Mio ou de `3 GB` en Go. Vous pouvez spécifier les valeurs de l'UC dans le fichier JSON sous forme de chaîne en unités d'UC ou en processeurs virtuels (vCPU). Par exemple, vous pouvez spécifier une valeur d'UC de `1024` en unités d'UC ou de `1 vCPU` en vCPU.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16 processeurs virtuels)	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure		
32768 (32 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	60 GO, 120 GO, 244 GO	Linux

Mise en réseau des tâches

Les tâches Amazon ECS pour AWS Fargate nécessitent le mode réseau `awsvpc`, qui fournit à chaque tâche une interface réseau Elastic. Lorsque vous exécutez une tâche ou créez un service avec ce mode réseau, vous devez spécifier un ou plusieurs sous-réseaux pour attacher l'interface réseau et un ou plusieurs groupes de sécurité à appliquer à l'interface réseau.

Si vous utilisez des sous-réseaux publics, déterminez s'il est nécessaire de fournir une adresse IP publique pour l'interface réseau. Pour qu'une tâche Fargate dans un sous-réseau public puisse extraire des images des conteneurs, une adresse IP publique doit être attribuée à l'interface réseau Elastic de la tâche, avec une route vers Internet ou une passerelle NAT pouvant acheminer les demandes vers Internet. Le sous-réseau nécessite qu'une passerelle NAT soit attachée aux demandes d'acheminement à Internet pour qu'une tâche Fargate dans un sous-réseau privé puisse extraire des images de conteneur. Lorsque vous hébergez vos images de conteneur dans Amazon ECR, vous pouvez configurer Amazon ECR pour utiliser un point de terminaison VPC d'interface. Dans ce cas, l'adresse IPv4 privée de la tâche est utilisée pour l'extraction de l'image. Pour plus d'informations sur les points de terminaison d'interface Amazon ECR, consultez Points

de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Voici un exemple de section `networkConfiguration` pour un service Fargate :

```
"networkConfiguration": {
   "awsvpcConfiguration": {
      "assignPublicIp": "ENABLED",
      "securityGroups": [ "sg-12345678" ],
      "subnets": [ "subnet-12345678" ]
   }
}
```

Limites des ressources de tâche

Les définitions de tâche Amazon ECS pour Linux sur AWS Fargate prennent en charge le paramètre `ulimits` permettant de définir les limites de ressources d'un conteneur.

Les définitions de tâche Amazon ECS pour Windows sur AWS Fargate ne prennent pas en charge le paramètre `ulimits` permettant de définir les limites de ressources d'un conteneur.

Les tâches Amazon ECS hébergées sur Fargate utilisent les valeurs de limite définies par défaut par le système d'exploitation, à l'exception du paramètre de limite de ressources `nofile`. La limite de ressources `nofile` restreint le nombre de fichiers ouverts qu'un conteneur peut utiliser. Sur Fargate, la limite flexible par défaut `nofile` est `65535` et la limite stricte est `65535`. Vous pouvez définir les valeurs des deux limites jusqu'à `1048576`.

L'exemple suivant présente un extrait de définition de tâche qui montre comment définir une limite `nofile` personnalisée qui a été doublée :

```
"ulimits": [
    {
       "name": "nofile",
       "softLimit": 2048,
       "hardLimit": 8192
    }
]
```

Pour plus d'informations sur les autres limites de ressources pouvant être ajustées, consultez Limites des ressources.

Logging

Journalisation des événements

Amazon ECS enregistre les actions qu'il entreprend pour EventBridge. Vous pouvez utiliser les événements Amazon ECS pour EventBridge pour recevoir des notifications en temps quasi réel concernant l'état actuel de vos clusters, services et tâches Amazon ECS. Vous pouvez également automatiser des actions pour répondre à ces événements. Pour plus d'informations, consultez Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge.

Journalisation du cycle de vie des tâches

Les tâches exécutées sur Fargate publient des horodatages pour suivre l'état de la tâche au cours de son cycle de vie. Vous pouvez voir les horodatages dans les détails des tâches dans le Console de gestion AWS et en décrivant la tâche dans le AWS CLI et des kits de développement logiciel. Par exemple, vous pouvez utiliser les horodatages pour évaluer le temps passé par la tâche à télécharger les images du conteneur et décider si vous devez optimiser la taille de l'image du conteneur ou utiliser des index Seekable OCI. Pour plus d'informations sur les pratiques relatives aux images de conteneur, veuillez consulter Bonnes pratiques en matière d'images de conteneurs Amazon ECS.

Journalisation d'applications

Les définitions de tâche Amazon ECS pour AWS Fargate prennent en charge uniquement les pilotes de journal `awslogs`, `splunk` et `awsfirelens` pour la configuration de journal.

Le `awslogs` log driver configure vos tâches Fargate pour envoyer les informations de journal à Amazon CloudWatch Journaux. L'exemple suivant montre un extrait d'une définition de tâche dans lequel le pilote de journal `awslogs` est configuré :

```
"logConfiguration": {
   "logDriver": "awslogs",
   "options": {
      "awslogs-group" : "/ecs/fargate-task-definition",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
   }
}
```

Pour plus d'informations sur l'utilisation du `awslogs` enregistrez le pilote dans une définition de tâche à laquelle envoyer les journaux de votre conteneur CloudWatch Journaux, voir Envoyer les journaux Amazon ECS à CloudWatch.

Pour plus d'informations sur l'utilisation du pilote de journal `awsfirelens` dans une définition de tâche, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

Pour plus d'informations sur l'utilisation du pilote de journal `splunk` dans une définition de tâche, consultez `splunk` pilote de journal.

Stockage de tâche

Pour les tâches Amazon ECS hébergées sur Fargate, les types de stockage suivants sont pris en charge :

- Les volumes Amazon EBS offrent un stockage par blocs économique, durable et hautement performant pour les charges de travail conteneurisées gourmandes en données. Pour plus d'informations, consultez Utilisation des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS pour le stockage permanent. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.
- Supports de liaison pour un stockage éphémère. Pour plus d'informations, consultez Utilisation de montages liés avec Amazon ECS.

Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI)

Les tâches Amazon ECS sur Fargate qui utilisent la version de plateforme `1.4.0` de Linux peuvent utiliser Seekable OCI (SOCI) pour accélérer le démarrage des tâches. Avec SOCI, les conteneurs ne passent que quelques secondes à extraire l'image avant de pouvoir démarrer, ce qui laisse le temps de configurer l'environnement et d'instancier l'application pendant que l'image est téléchargée en arrière-plan. C'est ce qu'on appelle le chargement différé. Lorsque Fargate lance une tâche Amazon ECS, il détecte automatiquement si un index SOCI existe pour une image de la tâche et démarre le conteneur sans attendre que l'image complète soit téléchargée.

Pour les conteneurs qui s'exécutent sans index SOCI, les images des conteneurs sont entièrement téléchargées avant le démarrage du conteneur. Ce comportement est le même sur toutes les autres versions de plateforme de Fargate et sur Amazon ECS-optimized AMI sur les instances Amazon EC2.

Seekable OCI (SOCI) est une technologie open source développée par AWS qui peut lancer des conteneurs plus rapidement en chargeant paresseusement l'image du conteneur. SOCI fonctionne en créant un index (index SOCI) des fichiers dans une image de conteneur existante. Cet index permet de lancer les conteneurs plus rapidement, en offrant la possibilité d'extraire un fichier individuel

d'une image de conteneur avant de télécharger l'image complète. L'index SOCI doit être stocké sous forme d'artefact dans le même référentiel que l'image dans le registre des conteneurs. Vous ne devez utiliser que les index SOCI provenant de sources fiables, car l'index est la source officielle du contenu de l'image. Pour plus d'informations, veuillez consulter Présentation de Seekable OCI pour les images de conteneur à chargement différé.

Les clients souhaitant utiliser SOCI ne pourront utiliser que le manifeste v2 d'index SOCI. Les clients existants qui ont déjà utilisé SOCI sur Fargate peuvent continuer à utiliser le manifeste v1 d'index SOCI mais nous leur conseillons vivement de passer à le manifeste v2 d'index SOCI. Le manifeste v2 d'index SOCI crée une relation explicite entre les images de conteneurs et leurs index SOCI afin de garantir des déploiements cohérents.

Considérations

Si vous souhaitez que Fargate utilise un index SOCI pour charger des images de conteneur de manière différée dans une tâche, prenez en compte les éléments suivants :

- Seules les tâches exécutées sur une version de plateforme Linux `1.4.0` peuvent utiliser les index SOCI. Les tâches exécutées sur des conteneurs Windows sur Fargate ne sont pas prises en charge.
- Les tâches exécutées sur une architecture de processeur X86_64 ou ARM64 sont prises en charge.
- Les images de conteneur incluses dans la définition de tâche doivent être stockées dans un registre d'images compatible. Voici une liste des registres compatibles :
  - Registres privés Amazon ECR.
- Seules les images de conteneur qui utilisent la compression gzip ou qui ne sont pas compressées sont prises en charge. Les images de conteneur qui utilisent la compression zstd ne sont pas prises en charge.
- Pour le manifeste v2 d'index SOCI, la génération d'un manifeste d'index SOCI modifie le manifeste de l'image du conteneur lorsque nous ajoutons une annotation pour l'index SOCI. Il en résulte un nouveau condensé d'image du conteneur. Le contenu des couches du système de fichiers des images de conteneur ne change pas.
- Pour le manifeste d'index SOCI v2, lorsque l'image de conteneur a déjà été stockée dans le référentiel d'images de conteneur, une fois qu'un index SOCI a été généré, vous devez republier l'image de conteneur. Re-pushing une image de conteneur n'augmentera pas les coûts de stockage en dupliquant les couches du système de fichiers, elle télécharge uniquement un nouveau fichier manifeste.

- Nous vous recommandons d'essayer le chargement différé avec des images de conteneur dont la taille est supérieure à 250 MiB compressés. Il est moins probable que le temps nécessaire pour charger des images plus petites soit réduit.
- Le chargement différé pouvant modifier le temps de démarrage de vos tâches, vous devrez peut-être modifier différents délais, tels que le délai de grâce de surveillance de l'état pour Elastic Load Balancing.
- Si vous souhaitez empêcher le chargement différé d'une image de conteneur, l'image de conteneur doit être republiée sans aucun index SOCI joint.

Création d'un index Seekable OCI

Pour qu'une image de conteneur soit chargée en différé, un index SOCI (un fichier de métadonnées) doit être créé et stocké dans le référentiel d'images de conteneur à côté de l'image de conteneur. Pour créer et publier un index SOCI, vous pouvez utiliser l'open source outil CLI soci-snapshotter sur GitHub. Vous pouvez également déployer le CloudFormation AWS Générateur d'index SOCI. Il s'agit d'une solution sans serveur qui crée et transmet automatiquement un index SOCI lorsqu'une image de conteneur est transmise à Amazon ECR. Pour plus d'informations sur la solution et les étapes d'installation, voir CloudFormation AWS Générateur d'index SOCI sur GitHub. Le CloudFormation AWS SOCI Index Builder est un moyen d'automatiser la mise en route avec SOCI, tandis que l'outil open source SOCI offre une plus grande flexibilité en matière de génération d'index et la possibilité d'intégrer la génération d'index dans votre intégration et votre livraison continues (CI/CD) canalisations.

> **Note**
>
> Pour que l'index SOCI soit créé pour une image, celle-ci doit exister dans le magasin d'images containerd de l'ordinateur exécutant `soci-snapshotter`. Si l'image se trouve dans le magasin d'images Docker, elle est introuvable.

Vérification de l'utilisation du chargement différé par une tâche

Pour vérifier qu'une tâche a été chargée en différé à l'aide de SOCI, vérifiez le point de terminaison des métadonnées de la tâche depuis cette dernière. Lorsque vous interrogez le point de terminaison des métadonnées de tâche version 4, il existe un champ `Snapshotter` dans le chemin par défaut du conteneur à partir duquel vous interrogez. De plus, il existe des champs `Snapshotter` pour chaque conteneur du chemin `/task`. La valeur par défaut de ce champ est `overlayfs`, et ce champ est

défini sur `soci` si SOCI est utilisé. Pour vérifier que le manifeste v2 d'index SOCI est attaché à une image de conteneur, vous pouvez récupérer l'index d'image auprès d'Amazon ECR à l'aide de l' AWS CLI.

```
IMAGE_REPOSITORY=r
IMAGE_TAG=latest

aws ecr batch-get-image \
    --repository-name=$IMAGE_REPOSITORY \
    --image-ids imageTag=$IMAGE_TAG \
    --query 'images[0].imageManifest' --output text | jq -r '.manifests[] |
 select(.artifactType=="application/vnd.amazon.soci.index.v2+json")'
```

Pour vérifier que le manifeste v1 d'index SOCI est attaché à une image de conteneur, vous pouvez utiliser l'API OCI Referrers.

```
ACCOUNT_ID=111222333444
AWS_REGION=us-east-1
IMAGE_REPOSITORY=nginx-demo
IMAGE_TAG=latest
IMAGE_DIGEST=$(aws ecr describe-images --repository-name $IMAGE_REPOSITORY --image-ids
 imageTag=$IMAGE_TAG --query 'imageDetails[0].imageDigest' --output text)
ECR_PASSWORD=$(aws ecr get-login-password)

curl \
    --silent \
    --user AWS:$ECR_PASSWORD \
    https://$ACCOUNT_ID.dkr.ecr.$AWS_REGION.amazonaws.com/v2/$IMAGE_REPOSITORY/
referrers/$IMAGE_DIGEST?artifactType=application%2Fvnd.amazon.soci.index.v1%2Bjson | jq
 -r '.'
```

Différences entre les définitions de tâche Amazon ECS pour les instances EC2 exécutant Windows

Les tâches exécutées sur les instances Windows EC2 ne prennent pas en charge tous les paramètres de définition de tâche Amazon ECS disponibles. Certains paramètres ne sont pas pris en charge du tout, tandis que d'autres se comportent différemment.

Les paramètres de définition de tâche suivants ne sont pas pris en charge pour les définitions de tâche Amazon EC2 Windows :

- `containerDefinitions`
  - `disableNetworking`
  - `dnsServers`
  - `dnsSearchDomains`
  - `extraHosts`
  - `links`
  - `linuxParameters`
  - `privileged`
  - `readonlyRootFilesystem`
  - `user`
  - `ulimits`
- `volumes`
  - `dockerVolumeConfiguration`
- `cpu`

  Nous vous recommandons de spécifier une UC de niveau conteneur pour les conteneurs Windows.
- `memory`

  Nous vous recommandons de spécifier une mémoire de niveau conteneur pour les conteneurs Windows.
- `proxyConfiguration`
- `ipcMode`
- `pidMode`
- `taskRoleArn`

  Les rôles IAM pour les tâches sur les instances Windows EC2 nécessitent une configuration supplémentaire, mais cette configuration est en grande partie similaire à la configuration des rôles IAM pour les tâches sur les instances de conteneurs Linux. Pour plus d'informations, consultez the section called “Configuration supplémentaire d'instance Windows Amazon EC2”.

Création d’une définition de tâche Amazon ECS à l’aide de la console

Vous créez une définition de tâche afin de pouvoir définir l’application que vous exécutez en tant que tâche ou service.

Lorsque vous créez une définition de tâche pour le type de lancement externe, vous devez créer la définition de tâche à l’aide de l’éditeur JSON et définir le `requiresCompatibilities` paramètre sur `EXTERNAL`.

Vous pouvez créer une définition de tâche en utilisant l’expérience de console ou en spécifiant un fichier JSON. Vous pouvez demander à Amazon Q de fournir des recommandations lorsque vous utilisez l’éditeur JSON. Pour plus d’informations, consultez Utilisation d’Amazon Q Developer pour proposer des recommandations de définition de tâche dans la console Amazon ECS

Validation JSON

L'éditeur JSON de la console Amazon ECS valide les éléments suivants dans le fichier JSON :

- Le fichier est un fichier JSON valide.
- Le fichier ne contient aucune clé superflue.
- Le fichier contient le paramètre `familyName`.
- Il y a au moins une entrée sous `containerDefinitions`.

CloudFormation piles

Le comportement suivant s’applique aux définitions de tâches qui ont été créées dans la nouvelle console avant le 12 janvier 2023.

Lorsque vous créez une définition de tâche, la console Amazon ECS crée automatiquement un CloudFormation pile dont le nom commence par `ECS-Console-V2-TaskDefinition-`. Si vous avez utilisé le AWS CLI ou un AWS SDK pour désenregistrer la définition de tâche, vous devez supprimer manuellement la pile de définitions de tâches. Pour plus d’informations, consultez la section Suppression d’une pile dans le Guide de l’utilisateur CloudFormation .

Les définitions de tâches créées après le 12 janvier 2023 n'ont pas de CloudFormation pile créée automatiquement pour eux.

Procédure

Amazon ECS console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Dans le menu Créer une définition de tâche, choisissez Créer une définition de tâche.
4. Pour Task definition family (Famille de définition de tâche), spécifiez un nom unique pour la définition de tâche.
5. Pour Type de lancement, choisissez l'environnement de l'application. La console par défaut est AWS Fargate (qui est sans serveur). Amazon ECS utilise cette valeur pour effectuer une validation afin de s’assurer que les paramètres de définition de tâche sont valides pour le type d’infrastructure.
6. Pour Fonctionnement system/Architecture, choisissez le système d'exploitation et l'architecture du processeur pour la tâche.

   Pour exécuter votre tâche sur une architecture ARM 64 bits, choisissez Linux/ARM64. Pour plus d’informations, consultez the section called “Plateforme d'exécution”.

   Pour exécuter vos tâches AWS Fargate sur les conteneurs Windows, choisissez un système d'exploitation Windows pris en charge. Pour plus d’informations, consultez the section called “Systèmes d’exploitation et architectures”.
7. Pour Task size (Taille de la tâche), choisissez les valeurs du processeur et de la mémoire à réserver pour la tâche. La valeur du processeur est indiquée en vCPU et la mémoire est indiquée en Go.

   Pour les tâches hébergées sur Fargate, le tableau suivant indique les combinaisons de processeur et de mémoire valides.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
32768 (32 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	60 GO, 120 GO, 244 GO	Linux

Pour les tâches qui utilisent des instances EC2 ou des instances externes, les valeurs UC prises en charge sont comprises entre 128 unités d'UC (0,125 vCPU) et 196 608 unités d'UC (192 vCPU).

Pour spécifier la valeur de la mémoire en Go, saisissez Go après la valeur. Par exemple, pour définir la valeur de la mémoire sur 3 Go, saisissez 3 Go.

> **Note**
>
> Task-level Les paramètres du processeur et de la mémoire sont ignorés pour les conteneurs Windows.

8. Pour Network mode (Mode réseau), choisissez le mode réseau à utiliser. La valeur par défaut est le mode awsvpc. Pour plus d'informations, consultez Réseaux des tâches Amazon ECS.

   Si vous choisissez pont, sous Mappages de port, pour Port hôte, saisissez le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

9. (Facultatif) Développez le Rôles des tâches section pour configurer le Gestion des identités et des accès AWS rôles (IAM) pour la tâche :

   a. Pour Task role (Rôle de tâche), choisissez le rôle IAM à assigner à la tâche. Le rôle IAM d'une tâche fournit des autorisations pour les conteneurs d'une tâche à appeler AWS Opérations d'API.

b. Pour Rôle d'exécution de tâche, choisissez le rôle.

   Pour savoir quand utiliser un rôle d'exécution de tâche, veuillez consulter the section called “Rôle IAM d'exécution de tâche”. Si vous n’avez pas besoin du rôle, choisissez Aucun.

10. (Facultatif) Développez la section Placement de tâche pour ajouter des contraintes de placement. Les contraintes de placement de tâche vous permettent de filtrer les instances de conteneur utilisées pour le placement de vos tâches à l’aide d’attributs intégrés ou personnalisés.
11. (Facultatif) Développez la section Injection de pannes pour activer l’injection de pannes. L’injection de pannes vous permet de tester la façon dont votre application réagit à certains scénarios de défaillance.
12. Pour chaque conteneur à définir dans votre définition de tâche, effectuez les étapes suivantes.

    a. Pour Name (Nom), saisissez un nom pour le conteneur.

    b. Pour Image URI (URI de l'image), saisissez l'image à utiliser pour démarrer un conteneur. Les images de la galerie publique Amazon ECR peuvent être spécifiées uniquement à l’aide du nom de registre Amazon ECR public. Par exemple, si `public.ecr.aws/ecs/amazon-ecs-agent:latest` est spécifié, le conteneur Amazon Linux hébergé dans la galerie publique Amazon ECR est utilisé. Pour tous les autres référentiels, spécifiez le référentiel en utilisant les formats `repository-url/image:tag` ou `repository-url/image@digest`.

    c. Si votre image se trouve dans un registre privé en dehors d'Amazon ECR, sous Registre privé, activez Authentification du registre privé. Ensuite, dans ARN ou nom Secrets Manager, saisissez l'Amazon Resource Name (ARN) du secret.

    d. Pour Conteneur essentiel, si deux conteneurs ou plus sont définis dans votre définition de tâche, vous pouvez spécifier si le conteneur doit être considéré comme essentiel. Si un conteneur est marqué comme Essentiel et qu’il s’arrête, la tâche est interrompue. Chaque définition de tâche doit contenir au moins un conteneur essentiel.

    e. Un mappage de ports permet aux conteneurs d'accéder aux ports de l'hôte pour envoyer ou recevoir du trafic. Sous Port mappings (Mappages de port), effectuez l'une des actions suivantes :

       - Lorsque vous utilisez le mode réseau awsvpc, pourContainer port (Port du conteneur) et Protocol (Protocole), choisissez le mappage de port à utiliser pour le conteneur.

- Lorsque vous utilisez le mode réseau bridge (pont), pour Container port (Port du conteneur) et Protocol (Protocole), choisissez le mappage de port à utiliser pour le conteneur.

Choisissez Add more port mappings (Ajouter d'autres mappages de ports) pour spécifier des mappages de ports de conteneur supplémentaires.

f. Pour donner au conteneur un accès en lecture seule à son système de fichiers racine, pour Système de fichiers racine en lecture seule, sélectionnez Lecture seule.

g. (Facultatif) Pour définir les limites d'UC, de GPU et de mémoire au niveau du conteneur qui sont différentes des valeurs au niveau des tâches sous Limites d'allocation de ressources, procédez comme suit :

- Dans UC), saisissez le nombre d'UC que l'agent de conteneur Amazon ECS réserve pour le conteneur.
- Dans GPU, entrez le nombre d'unités GPU pour l'instance de conteneur.

  Une instance Amazon EC2 prenant en charge les GPU possède une unité GPU pour chaque GPU. Pour plus d'informations, consultez the section called "Définitions de tâches pour les charges de travail GPU".
- Pour Limite stricte de mémoire, saisissez la quantité de mémoire, en Go, à attribuer au conteneur. Si le conteneur tente de dépasser la limite stricte, il s'arrête.
- Le démon Docker 20.10.0 ou version ultérieure réserve un minimum de 6 Mio de mémoire pour un conteneur. Vous ne devez donc pas spécifier moins de 6 Mio de mémoire pour vos conteneurs.

  Le démon Docker 19.03.13-ce ou version antérieure réserve un minimum de 4 Mio de mémoire pour un conteneur. Vous ne devez donc pas spécifier moins de 4 Mio de mémoire pour vos conteneurs.
- Pour Limite flexible de mémoire, saisissez la limite flexible (en Go) de mémoire à réserver pour le conteneur.

  En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Si vous ne spécifiez pas de mémoire au niveau de la tâche, vous devez indiquer un nombre entier différent de zéro pour Limite stricte de mémoire et/ou Limite flexible de mémoire. Si vous spécifiez les deux, Limite stricte de mémoire doit être supérieure à Limite flexible de mémoire.

Cette fonctionnalité n’est pas prise en charge sur les conteneurs Windows.

h. (Facultatif) Développez la section Variables d'environnement pour spécifier les variables d'environnement à injecter dans le conteneur. Vous pouvez spécifier des variables d’environnement soit individuellement à l’aide de paires clé-valeur, soit en bloc en spécifiant un fichier de variables d’environnement hébergé dans un compartiment Amazon S3. Pour plus d’informations sur le formatage d’un fichier de variable d’environnement, consultez la section Transmission d’une variable d’environnement individuelle à un conteneur Amazon ECS.

   Lorsque vous spécifiez une variable d’environnement pour le stockage des secrets, saisissez le nom du secret dans le champ Clé. Puis pour ValueFrom, entrez l'ARN complet du secret du magasin de paramètres de Systems Manager ou du secret du gestionnaire de secrets

i. (Facultatif) Sélectionnez l'option Use log collection (Utiliser la collecte de journaux) pour spécifier une configuration de journal. Pour chaque pilote de journal disponible, il existe des options de pilote de journal à spécifier. L'option par défaut envoie les journaux des conteneurs à Amazon CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens. Pour plus d’informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Exporter des journaux vers Splunk : configurez la tâche pour envoyer les journaux des conteneurs au pilote Splunk qui envoie les journaux à un service distant. Vous devez saisir l’URL de votre service Web Splunk. Le paramètre de jeton Splunk est spécifiée en tant qu’option secrète, car il peut être traité comme des données sensibles.
   - Exporter des journaux vers Amazon Data Firehose : configurez la tâche pour envoyer les journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Les journaux sont alors envoyés vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.

- Exporter des journaux vers Amazon Kinesis Data Streams : configurez la tâche pour envoyer les journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Les journaux sont alors envoyés vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
- Exporter les journaux vers Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
- Exporter des journaux vers Amazon S3 : configurez la tâche pour envoyer les journaux de conteneur à un compartiment Amazon S3. Les options de pilote de journal par défaut sont fournies, mais vous devez spécifier un nom de compartiment Simple Storage Service (Amazon S3) valide.

j. (Facultatif) Configurez des paramètres de conteneur supplémentaires.

Pour configurer cette option	Faites ceci	
Politique de redémarrage Ces options définissent une politique de redémarrage pour redémarrer un conteneur lorsqu'il se ferme.	Développez Politique de redémarrage, puis configurez les éléments suivants : • Pour activer une politique de redémarrage pour le conteneur, activez l'option Activer la politique de redémarrage. • Pour Codes de sortie ignorés, spécifiez une liste séparée par des virgules de codes de sortie de conteneur entiers. Si le conteneur se ferme avec l'un des codes de sortie spécifiés, Amazon ECS ne tentera pas de redémarrer le conteneur. Si rien n'est spécifié, Amazon ECS n'ignorera aucun code de sortie. • Pour Période de tentative de réinitialisation, spécifiez une période entière, en secondes, pendant	

Pour configurer cette option	Faites ceci	
	laquelle le conteneur doit fonctionner avant de pouvoir tenter de redémarrer en cas de sortie. Amazon ECS ne peut tenter de redémarrer un conteneur qu'une seule fois par période de tentative de réinitialisation (secondes). Si rien n'est spécifié, le conteneur doit fonctionner pendant 300 secondes avant toute tentative de redémarrage.	

Pour configurer cette option	Faites ceci	
HealthCheck Ce sont les commandes qui déterminent si un conteneur est sain. Pour plus d'informations, consultez Détermination de l'état des tâches Amazon ECS à l'aide de la surveillance de l'état des conteneurs.	Élargir HealthCheck, puis configurez les éléments suivants : • Pour Command (Commande), saisissez une liste de commandes séparées par des virgules. Vous pouvez démarrer les commandes par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé. • Pour Interval (Interval le), entrez le nombre de secondes entre chaque surveillance de l'état. Les valeurs valides sont comprises entre 5 et 30. • Pour Timeout (Délai), entrez la durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant	

Pour configurer cette option	Faites ceci	
	qu'elle ne soit considérée comme un échec. Les valeurs valides sont comprises entre 2 et 60. • Pour Start period (Période de démarrage), entrez la durée (en secondes) d'attente pour qu'un conteneur s'amorce avant que les commandes de surveillance de l'état s'exécutent. Les valeurs valides sont comprises entre 0 et 300. • Pour Retries (Tentatives), entrez le nombre de tentatives d'exécution des commandes de surveillance de l'état en cas d'échec. Les valeurs valides sont comprises entre 1 et 10.	

Pour configurer cette option	Faites ceci	
Ordre des dépendances au démarrage Cette option définit les dépendances pour le démarrage et l'arrêt des conteneurs. Un conteneur peut contenir plusieurs dépendances.	Développez Ordre des dépendances de démarrage, puis configurez les éléments suivants : a. Choisissez Ajouter une dépendance de conteneur. b. Pour Conteneur, choisissez le conteneur. c. Pour Condition, choisissez la condition de dépendance de démarrage. Pour ajouter une dépendance supplémentaire, choisissez Ajouter une dépendance de conteneur.	

Pour configurer cette option	Faites ceci	
Délais de conteneur Ces options déterminent à quel moment démarrer et arrêter un conteneur.	Développez Délais de conteneur, puis configurez les éléments suivants : • Pour configurer le délai d'attente avant l'abandon de la résolution des dépendances pour un conteneur, saisissez le nombre de secondes dans Délai de démarrage. • Pour configurer le délai d'attente avant l'arrêt du conteneur s'il ne se ferme pas normalement tout seul, saisissez le nombre de secondes dans Délai d'arrêt.	

Pour configurer cette option	Faites ceci
Paramètres réseau de conteneur Ces options déterminent s'il faut utiliser la mise en réseau au sein d'un conteneur.	Développez Paramètres réseau de conteneur, puis configurez les éléments suivants : • Pour désactiver la mise en réseau des conteneurs, sélectionnez Désactiver le réseau. • Pour configurer les adresses IP des serveurs DNS présentées au conteneur, dans Serveurs DNS, saisissez l'adresse IP de chaque serveur sur une ligne distincte. • Pour configurer les domaines DNS afin de rechercher les noms d’hôtes non entièrement qualifiés présentés au conteneur, dans Domaines de recherche DNS, saisissez chaque domaine sur une ligne distincte.

Pour configurer cette option	Faites ceci	
	Le schéma est **^[a-zA-Z0-9-.]{0,253}[a-zA-Z0-9]$** . • Pour configurer le nom d'hôte du conteneur , dans Nom d'hôte, saisissez le nom d'hôte du conteneur. • Pour ajouter des noms d'hôte et des mappages d'adresses IP ajoutés au fichier `/etc/hosts` sur le conteneur, choisissez Ajouter un hôte supplémentaire, puis pour Nom d'hôte et Adresse IP, saisissez le nom d'hôte et l'adresse IP.	

Pour configurer cette option	Faites ceci	
Configuration Docker Elles remplacent les valeurs dans le Dockerfile.	Développez Configuration Docker, puis configurez les éléments suivants : • Pour Commande, saisissez une commande exécutable pour un conteneur. Ce paramètre correspond à `Cmd` dans la section Créer un conteneur de l'API Docker Remote et à l'option `COMMAND` dans `docker run`. Ce paramètre remplace l'instruction `CMD` contenue dans un Dockerfile. • Pour Point d'entrée, saisissez l'ENTRYPOINT Docker qui est transmis au conteneur. Ce paramètre correspond à `Entrypoint` dans la section Créer un conteneur de l'API Docker Remote et à l'option `--entrypoint` dans `docker run`. Ce paramètre	

Pour configurer cette option	Faites ceci	
	remplace l'instruction `ENTRYPOINT` contenue dans un Dockerfile. • Dans le champ Répertoire de travail, entrez le répertoire dans lequel le conteneur exécutera tout point d'entrée et les instructions de commande fournies. Ce paramètre correspond à `WorkingDir` dans la section Créer un conteneur de l'API Docker Remote et à l'option `--workdir` dans `docker run` . Ce paramètre remplace l'instruction `WORKDIR` contenue dans un Dockerfile.	

Pour configurer cette option	Faites ceci
Limites des ressources (Ulimits) Ces valeurs remplacent le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond à `Ulimits` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--ulimit` correspond à docker run.	Développez Limites de ressources (ulimits), puis choisissez Ajouter ulimit. Pour Nom de limite, choisissez la limite. Ensuite, pour Limite flexible et Limite stricte, saisissez les valeurs. Pour en ajouter d'autres ulimits, choisissez Ajouter ulimit.
Docker étiquettes Cette option ajoute des métadonnées à votre conteneur. Ce paramètre correspond à `Labels` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--label` correspond à docker run.	Développez Étiquettes Docker, choisissez Ajouter une paire clé-valeur, puis saisissez la Clé et la Valeur. Pour ajouter des étiquettes Docker supplémentaires, choisissez Ajouter une paire clé-valeur.

k. (Facultatif) Choisissez Add more containers (Ajouter des conteneurs supplémentaires) pour ajouter des conteneurs supplémentaires à la définition de tâche.

13. (Facultatif) La section Stockage permet d'étendre la quantité de magasins éphémères pour les tâches hébergées sur Fargate. Vous pouvez également utiliser cette section pour ajouter une configuration de volume de données à la tâche.

    - Pour étendre le stockage éphémère disponible au-delà de la valeur par défaut de 20 gibibytes (GiB) pour vos tâches Fargate pour Amount (Quantité), spécifiez une valeur allant jusqu'à 200 GiB.

14. (Facultatif) Pour ajouter une configuration de volume de données à la définition de tâche, sélectionnez Ajouter un volume, puis procédez comme suit.

    a. Pour Volume name (Nom du volume), saisissez un nom pour le volume de données. Le nom du volume de données est utilisé lors de la création d'un point de montage de conteneur.

    b. Pour Configuration du volume, indiquez si vous souhaitez configurer votre volume lors de la création de la définition de tâche ou lors du déploiement.

    > **Note**
    >
    > Les volumes pouvant être configurés lors de la création d'une définition de tâche comprennent les montages liés, Docker, Amazon EFS et Amazon FSx for Windows File Server. Les volumes qui peuvent être configurés lors du déploiement lors de l'exécution d'une tâche, ou lors de la création ou de la mise à jour d'un service incluent Amazon EBS.

    c. Pour Type de volume, sélectionnez un type de volume compatible avec le type de configuration que vous avez sélectionné, puis configurez le type de volume.

Type de volume	Étapes	
Montage lié	a. Choisissez Add mount point (Ajouter un point de montage), puis configure z les éléments suivants : •	

Type de volume	Étapes	
	Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. b. Pour ajouter des points de montage supplémentaires, ajoutez un point de montage.	

Type de volume	Étapes	
EFS	a. Pour File system ID (ID du système de fichiers), choisissez l'ID du système de fichiers Amazon EFS. b. (Facultatif) Pour Root directory (Répertoire racine, entrez le répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. Si vous prévoyez d'utiliser un point d'accès EFS, laissez ce champ vide. c. (Facultatif) Pour Access point (Point d'accès), choisissez l'ID du point d'accès à utiliser. d. (Facultatif) Pour chiffrer les données entre le système de fichiers Amazon EFS et l'hôte Amazon ECS ou pour utiliser le rôle d'exécution de tâches lors du	

Type de volume	Étapes	
	montage du volume, choisissez Advanced configurations (Configurations avancées), puis configurez les éléments suivants : • Pour chiffrer les données entre le système de fichiers Amazon EFS et l'hôte Amazon ECS, sélectionnez Transit encryption (Chiffrement de transit), puis pour Port, entrez le port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, il utilise la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.	

Type de volume	Étapes	
	• Pour utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définitio n de tâche lors du montage du système de fichiers Amazon EFS, sélectionnez IAM authorization (Autorisa tion IAM). e. Choisissez Add mount point (Ajouter un point de montage), puis configure z les éléments suivants : • Sous Container (Conteneur), choisisse z le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. •	

Type de volume	Étapes	
	Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. f. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
Docker	a. Pour Pilote, saisissez la configuration du volume Docker. Les conteneurs Windows ne prennent en charge que l'utilisation du pilote local. Pour utiliser des montages bind, spécifiez un hôte. b. Pour Scope (Portée), choisissez le cycle de vie du volume. • Pour que le cycle de vie dure au démarrage et à l'arrêt de la tâche, choisissez Task (Tâche). • Pour que le volume persiste après l'arrêt de la tâche, choisissez Shared (Partagé). c. Choisissez Add mount point (Ajouter un point de montage), puis configurez les éléments suivants : • Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage.	

Type de volume	Étapes	
	• Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. d. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
FSx for Windows File Server	a. Dans ID de système de fichiers, choisissez l'ID de système de fichiers FSx for Windows File Server. b. Dans Répertoire racine, saisissez le répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l’intérieur de l’hôte. c. Dans Credential parameter (Paramètre d’informations d’identif ication), choisissez la manière dont les informations d'identif ication sont stockées. • À utiliser AWS Secrets Manager, entrez le nom de ressource Amazon (ARN) d'un secret de Secrets Manager. • À utiliser AWS Systems Manager, entrez le nom de ressource Amazon	

Type de volume	Étapes	
	(ARN) d'un paramètre de Systems Manager. d. Pour Domaine, saisissez le nom de domaine complet hébergé par un répertoire AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou un Active Directory EC2 autohébergé. e. Choisissez Add mount point (Ajouter un point de montage), puis configurez les éléments suivants : • Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume.	

Type de volume	Étapes	
	• Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. f. Pour ajouter des points de montage supplémen taires, ajoutez un point de montage.	

Type de volume	Étapes	
Amazon EBS	a. Choisissez Add mount point (Ajouter un point de montage), puis configurez les éléments suivants : • Sous Container (Conteneur), choisissez le conteneur correspondant au point de montage. • Pour Source volume (Volume source), choisissez le volume de données à monter sur le conteneur. • Sous Container path (Chemin du conteneur), saisissez le chemin d'accès au conteneur pour monter le volume. • Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume. b. Pour ajouter des points de montage supplémentaires, ajoutez un point de montage.	

Type de volume	Étapes	

15. Pour ajouter un volume depuis un autre conteneur, choisissez Ajouter un volume à partir de, puis configurez les éléments suivants :

    - Pour Conteneur, choisissez le conteneur.
    - Pour Source, choisissez le conteneur contenant le volume que vous souhaitez monter.
    - Pour Lecture seule, indiquez si le conteneur dispose d'un accès en lecture seule au volume.

16. (Facultatif) Pour configurer les paramètres de suivi et de collecte des métriques de votre application à l'aide du AWS Distro for OpenTelemetry intégration, expansion Surveillance, puis sélectionnez Utiliser la collecte de métriques pour collecter et envoyer des statistiques relatives à vos tâches à Amazon CloudWatch ou Amazon Managed Service pour Prometheus. Lorsque cette option est sélectionnée, Amazon ECS crée un AWS Distro for OpenTelemetry side-car de conteneur préconfiguré pour envoyer les métriques de l'application. Pour plus d'informations, consultez Corrélation des performances des applications Amazon ECS à l'aide des métriques d'application.

    a. Quand Amazon CloudWatch est sélectionné, les métriques de votre application personnalisée sont acheminées vers CloudWatch sous forme de métriques personnalisées. Pour plus d'informations, consultez Exporter les métriques des applications vers Amazon CloudWatch.

    > **Important**
    >
    > Lors de l'exportation de statistiques d'application vers Amazon CloudWatch, votre définition de tâche nécessite un rôle IAM de tâche avec les autorisations requises. Pour plus d'informations, consultez Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec Amazon CloudWatch.

    b. Lorsque vous sélectionnez Amazon Managed Service for Prometheus (Prometheus libraries instrumentation) (Amazon Managed Service for Prometheus [instrumentation de bibliothèques Prometheus]), vos métriques de processeur, de la mémoire, du réseau et du stockage au niveau des tâches et vos métriques d'application personnalisées sont acheminées vers Amazon Managed Service for Prometheus. Pour Point de

terminaison de l'écriture à distance de l'espace de travail, saisissez l'URL du point de terminaison d'écriture à distance pour votre espace de travail Prometheus. Pour Cible de récupération, saisissez l'hôte et le port que le collecteur AWS Distro for OpenTelemetry peut utiliser pour récupérer des données de métriques. Pour plus d'informations, consultez Exportation des métriques d'application vers Amazon Managed Service for Prometheus.

> **Important**
>
> Lorsque vous exportez des métriques d'application vers Amazon Managed Service for Prometheus, votre définition de tâche nécessite un rôle IAM de tâche avec les autorisations nécessaires. Pour plus d'informations, consultez Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec Amazon Managed Service pour Prometheus.

c. Lorsque vous sélectionnez Service géré Amazon pour Prometheus (OpenTelemetry instrumentation), vos mesures relatives au processeur, à la mémoire, au réseau et au stockage au niveau des tâches, ainsi que les mesures de vos applications personnalisées sont acheminées vers Amazon Managed Service for Prometheus. Pour Point de terminaison de l'écriture à distance de l'espace de travail, saisissez l'URL du point de terminaison d'écriture à distance pour votre espace de travail Prometheus. Pour plus d'informations, consultez Exportation des métriques d'application vers Amazon Managed Service for Prometheus.

> **Important**
>
> Lorsque vous exportez des métriques d'application vers Amazon Managed Service for Prometheus, votre définition de tâche nécessite un rôle IAM de tâche avec les autorisations nécessaires. Pour plus d'informations, consultez Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec Amazon Managed Service pour Prometheus.

17. (Facultatif) Développez la section Tags (Identifications) pour ajouter des identifications, sous forme de paires clé-valeur, à la définition de tâche.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.

- Pour Valeur, saisissez la valeur de clé.
- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

18. Choisissez Créer pour enregistrer la définition de tâche.

Amazon ECS console JSON editor

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Créer une définition de tâche, puis Créer une définition de tâche avec JSON.
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON,

   Le JSON doit réussir les contrôles de validation spécifiés dans the section called “Validation JSON”.
5. Cliquez sur Créer.

Utilisation d’Amazon Q Developer pour proposer des recommandations de définition de tâche dans la console Amazon ECS

Lorsque vous utilisez l'éditeur JSON de la console Amazon ECS pour créer une définition de tâche, vous pouvez utiliser Amazon Q Developer pour fournir AI-generated des suggestions de code pour vos définitions de tâches.

Vous pouvez utiliser la fonctionnalité de chat en ligne pour demander à Amazon Q Developer de générer, d’expliquer ou de refactoriser le JSON de définition des tâches à l’aide d’une interface conversationnelle. Vous pouvez injecter les suggestions générées à tout moment dans la définition de la tâche et accepter ou rejeter les modifications proposées. Amazon ECS a également amélioré la fonctionnalité de suggestions en ligne existante afin d’utiliser Amazon Q Developer.

Lorsque vous créez une définition de tâche à l’aide de l’éditeur JSON, Amazon Q Developer peut vous fournir des recommandations pour vous aider à créer une définition de tâche plus rapidement. Vous pouvez avoir des suggestions intégrées basées sur les propriétés ou utiliser les suggestions Amazon Q Developer pour compléter automatiquement des blocs entiers d’exemples de code.

Vous pouvez utiliser cette fonctionnalité dans les régions où Amazon Q Developer est pris en charge. Pour plus d'informations, consultez la section Services AWS par région.

Conditions préalables

Les conditions suivantes sont requises :

- En plus des autorisations de la console, l'utilisateur qui crée la définition de tâche dans la console doit disposer de l'autorisation `codewhisperer:GenerateRecommendations` pour les recommandations et `q:SendMessage` pour utiliser le chat en ligne. Pour plus d'informations, consultez Autorisations requises pour utiliser Amazon Q Developer afin de fournir des recommandations dans la console.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Créer une définition de tâche, puis Créer une définition de tâche avec JSON.

   La page Créer une définition de tâche s'ouvre.

   La console fournit le modèle par défaut suivant.

```
{
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "essential": true
        }
    ],
    "volumes": [],
    "networkMode": "awsvpc",
    "memory": "3 GB",
    "cpu": "1 vCPU",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
```

```
}
```

4. Dans la fenêtre contextuelle des suggestions intégrées d'Amazon Q, sélectionnez Autoriser.

   Si vous fermez la fenêtre contextuelle, vous pouvez activer Amazon Q sous l'icône représentant une roue dentée.

5. Dans la zone de l'éditeur JSON, modifiez le document JSON.

   Pour qu'Amazon Q crée et renseigne les paramètres, saisissez un commentaire contenant ce que vous souhaitez ajouter. Dans l'exemple suivant, le commentaire amène Amazon Q à générer les lignes en gras.

```
{
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "essential": true
        },
        // add an nginx container using an image from Public ECR, with port 80
 open, and send logs to CloudWatch log group "myproxy"
        {
            "name": "nginx",
            "image": "public.ecr.aws/nginx/nginx:latest",
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "myproxy",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "nginx"
                }
```

```
            }
        }

    ],
    "volumes": [],
    "networkMode": "awsvpc",
    "memory": "3 GB",
    "cpu": "1 vCPU",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

6. Pour utiliser la fonctionnalité de chat en ligne, vous pouvez surligner les lignes, puis choisir l'icône en forme d'étoile.

   La boîte de discussion Amazon Q Developer s'affiche.

   Saisissez votre requête.

   Amazon Q Developer génère puis met à jour le JSON.

   Pour accepter les modifications, choisissez Accepter tout
7. Cliquez sur Créer.

Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console

Une Révision de définition de tâche est une copie de la définition de tâche actuelle avec les nouvelles valeurs de paramètres remplaçant les valeurs existantes. Tous les paramètres que vous ne modifiez pas se trouvent dans la nouvelle révision.

Pour mettre à jour une définition de tâche, créez une révision de définition de tâche. Si la définition de tâche est utilisée dans un service, vous devez mettre à jour ce dernier de sorte à utiliser la définition de tâche actualisée.

Lorsque vous créez une révision, vous pouvez modifier les propriétés de conteneur et les propriétés d'environnement suivantes.

- URI de l'image de conteneur
- Cartographie des ports
- Variables d'environnement

- Exigences en matière d'infrastructure
- Taille de la tâche
- Taille du contenant
- Rôle de tâche
- Rôle d'exécution de tâche
- Volumes et points de montage des conteneurs
- Registre privé

Vous pouvez demander à Amazon Q de fournir des recommandations lorsque vous utilisez l'éditeur JSON. Pour plus d'informations, consultez Utilisation d'Amazon Q Developer pour proposer des recommandations de définition de tâche dans la console Amazon ECS

Validation JSON

L'éditeur JSON de la console Amazon ECS valide les éléments suivants dans le fichier JSON :

- Le fichier est un fichier JSON valide
- Le fichier ne contient aucune clé superflue
- Le fichier contient le paramètre `familyName`
- Il y a au moins une entrée sous `containerDefinitions`

Procédure

Amazon ECS console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Choisissez la définition de tâche.
5. Sélectionnez la révision de définition de tâche, puis choisissez Créer une nouvelle révision, Créer une nouvelle révision.
6. Sur la page Create new task definition revision (Créer une nouvelle révision de définition de tâche), effectuez les modifications souhaitées. Par exemple, si vous souhaitez modifier les définitions du conteneur existantes (par exemple, l'image de conteneur, les limites de

mémoire ou les mappages de ports), sélectionnez le conteneur et appliquez les modifications. Vous pouvez mettre à jour la compatibilité de la définition de tâche vers AWS Fargate, Instances gérées ou Instances Amazon EC2.

7. Vérifiez les informations, puis choisissez Mettre à jour.
8. Si votre définition de tâche est utilisée dans un service, mettez à jour votre service avec la définition de tâche mise à jour. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Amazon ECS console JSON editor

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new revision (Créer une nouvelle révision), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON,

   Le JSON doit réussir les contrôles de validation spécifiés dans the section called "Validation JSON".
5. Cliquez sur Créer.

Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console

Vous pouvez annuler l'enregistrement de la révision de la définition de tâche afin qu'elle n'apparaisse plus dans vos appels API `ListTaskDefinition` ou dans la console lorsque vous souhaitez exécuter une tâche ou mettre à jour un service.

Lorsque vous annulez l'enregistrement de la révision d'une définition de tâche, celle-ci est immédiatement marquée comme `INACTIVE`. Les tâches et services existants qui font référence à une révision de définition de tâche `INACTIVE` continuent à s'exécuter sans interruption. Les services existants qui font référence à une révision de définition de tâche `INACTIVE` peuvent toujours être mis à l'échelle en augmentant ou réduisant le nombre de services souhaité.

Vous ne pouvez pas utiliser une révision de définition de tâche `INACTIVE` pour exécuter de nouvelles tâches ou créer de nouveaux services. Vous ne pouvez pas non plus mettre à jour un

service existant pour faire référence à une révision de définition de tâche `INACTIVE` (mais ces restrictions peuvent ne prendre effet que jusqu'à 10 minutes suivant l'annulation de l'enregistrement).

> **Note**
>
> Lorsque vous annulez l'enregistrement de toutes les révisions dans une famille de tâches, la famille de définition de tâche est déplacée vers la liste `INACTIVE`. L'ajout d'une nouvelle révision d'une définition de tâche `INACTIVE` ramène la famille de définition de tâche dans la liste `ACTIVE`.
>
> À l'heure actuelle, les révisions de définition de tâche `INACTIVE` restent détectables dans votre compte indéfiniment. Cependant, ce comportement est sujet à changement à l'avenir. Par conséquent, vous ne devriez pas compter sur une conservation de révisions de définition de tâche `INACTIVE` au-delà du cycle de vie des tâches et services associés.

CloudFormation piles

Le comportement suivant s’applique aux définitions de tâches qui ont été créées dans la nouvelle console avant le 12 janvier 2023.

Lorsque vous créez une définition de tâche, la console Amazon ECS crée automatiquement un CloudFormation pile dont le nom commence par `ECS-Console-V2-TaskDefinition-`. Si vous avez utilisé le AWS CLI ou un AWS SDK pour désenregistrer la définition de tâche, vous devez supprimer manuellement la pile de définitions de tâches. Pour plus d’informations, consultez la section Suppression d’une pile dans le Guide de l’utilisateur CloudFormation .

Les définitions de tâches créées après le 12 janvier 2023 n'ont pas de CloudFormation pile créée automatiquement pour eux.

Procédure

Annuler l'enregistrement d'une nouvelle définition de tâche (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Sur la page Task definitions (Définitions de tâche), choisissez la famille de la définition de tâche contenant une ou plusieurs révisions dont vous souhaitez annuler l'enregistrement.

5. Sur la page Nom de la définition de la tâche, sélectionnez les révisions à supprimer, puis choisissez Actions, Annuler l'enregistrement.
6. Vérifiez les informations contenues dans la fenêtre Deregister (Annuler l'enregistrement), puis choisissez Deregister (Annuler l'enregistrement) pour terminer.

Suppression de l’enregistrement d’une révision de définition de tâche Amazon ECS à l’aide de la console

Lorsque vous n’avez plus besoin d’une révision de définition de tâche spécifique dans Amazon ECS, vous pouvez supprimer la révision de définition de tâche.

Lorsque vous supprimez une révision de définition de tâche, elle passe immédiatement de l'état `INACTIVE` à l'état `DELETE_IN_PROGRESS`. Les tâches et services existants qui font référence à une révision de définition de tâche `DELETE_IN_PROGRESS` continuent à s'exécuter sans interruption.

Vous ne pouvez pas utiliser une révision de définition de tâche `DELETE_IN_PROGRESS` pour exécuter de nouvelles tâches ou créer de nouveaux services. Vous ne pouvez pas non plus mettre à jour un service existant pour référencer une révision de définition de tâche `DELETE_IN_PROGRESS`.

Lorsque vous supprimez toutes les révisions de définition de tâche `INACTIVE`, le nom de la définition de tâche n'est pas affiché dans la console et n'est pas renvoyé dans l'API. Si une révision de définition de tâche est à l’état `DELETE_IN_PROGRESS`, le nom de la définition de tâche est affiché dans la console et renvoyé dans l’API. Le nom de la définition de tâche est conservé par Amazon ECS et la révision est incrémentée la prochaine fois que vous créerez une définition de tâche portant ce nom.

Ressources Amazon ECS pouvant bloquer une suppression

Aucune requête de suppression de définition de tâche ne sera traitée tant qu’il existe des ressources Amazon ECS qui dépendent de la révision de la définition de tâche. Les ressources suivantes peuvent empêcher la suppression d'une définition de tâche :

- Tâches autonomes Amazon ECS : la définition de tâche est nécessaire pour que la tâche reste saine.
- Tâches d’un service Amazon ECS : la définition de tâche est nécessaire pour que la tâche reste saine.

- Déploiements et jeux de tâches d'un service Amazon ECS : la définition de tâche est nécessaire lorsqu'un événement de mise à l'échelle est lancé pour un déploiement ou un jeu de tâches Amazon ECS.

Si votre définition de tâche reste dans `DELETE_IN_PROGRESS` état, vous pouvez utiliser la console ou le AWS CLI pour identifier, puis arrêter les ressources qui bloquent la suppression de la définition de tâche.

Suppression de la définition de tâche après la suppression de la ressource bloquée

Les règles suivantes s'appliquent une fois que vous avez supprimé les ressources bloquant la suppression de la définition de tâche :

- Tâches Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à une heure après l'arrêt de la tâche.
- Déploiements et jeux de tâches d'un service Amazon ECS : la suppression de la définition de tâche peut prendre jusqu'à 24 heures après la suppression du déploiement ou du jeu de tâches.

Procédure

Pour supprimer les définitions de tâche (console Amazon ECS)

Vous devez annuler l'enregistrement d'une révision de définition de tâche avant de la supprimer. Pour plus d'informations, consultez the section called "Annulation de l'enregistrement d'une révision de définition de tâche à l'aide de la console".

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région qui contient votre définition de tâche.
3. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
4. Sur la page Task definitions (Définitions de tâche), choisissez la famille de la définition de tâche contenant une ou plusieurs révisions que vous souhaitez supprimer.
5. Sur la page Nom de la définition de la tâche, sélectionnez les révisions à supprimer, puis choisissez Actions, Supprimer.

   Si Supprimer n'est pas disponible, vous devez annuler l'enregistrement de la définition de tâche.
6. Vérifiez les informations contenues dans la fenêtre Supprimer, puis choisissez Supprimer pour terminer.

Cas d’utilisation de définition de tâche Amazon ECS

En savoir plus sur la rédaction de définitions de tâches pour différents AWS services et fonctionnalités.

En fonction de votre charge de travail, certains paramètres de définition des tâches doivent être définis. Pour EC2 également, vous devez choisir des instances spécifiques adaptées à la charge de travail.

Rubriques

- Définitions de tâches Amazon ECS pour les charges de travail GPU
- Définitions de tâche Amazon ECS pour les charges de travail de transcodage vidéo
- Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones
- Définitions de tâche Amazon ECS pour les instances de deep learning
- Définitions de tâche Amazon ECS pour les charges de travail ARM 64 bits
- Envoyer les journaux Amazon ECS à CloudWatch
- Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner
- En utilisant des produits non AWS images de conteneurs dans Amazon ECS
- Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l’aide de politiques de redémarrage de conteneurs
- Transmission de données sensibles vers un conteneur Amazon ECS

Définitions de tâches Amazon ECS pour les charges de travail GPU

Amazon ECS supporte les charges de travail utilisant des GPU lorsque vous créez des clusters avec des instances de conteneurs prenant en charge les GPU. Amazon EC2 GPU-based les instances de conteneur qui utilisent les types d'instance p2, p3, p4d, p5, g3, g4, g5, g6, g6e et g6f permettent d'accéder aux GPU NVIDIA. Pour plus d’informations, consultez la section Instances de calcul accélérées Linux du Guide des types d’instance Amazon EC2.

Amazon ECS fournit un GPU-optimized AMI fournie avec des pilotes de noyau NVIDIA préconfigurés et un moteur d'exécution pour GPU Docker. Pour plus d’informations, consultez Amazon ECS-optimized AMI Linux.

Vous pouvez désigner plusieurs GPU dans votre définition de tâche pour le placement des tâches au niveau d'un conteneur. Amazon ECS planifie les instances de conteneurs disponibles qui prennent en charge les GPU et associe les GPU physiques aux conteneurs appropriés pour des performances optimales.

L'Amazon EC2 suivant GPU-based les types d'instances sont pris en charge. Pour plus d'informations, voir Instances Amazon EC2 P2, Instances Amazon EC2 P3, Instances Amazon EC2 P4d, Instances Amazon EC2 P5, Instances Amazon EC2 G3, Instances Amazon EC2 G4, Instances Amazon EC2 G5, Instances Amazon EC2 G6, Instances Amazon EC2 G6e, et Instances Amazon EC2 G6f.

Type d'instance	GPU	Mémoire GPU (Gio)	vCPU	Mémoire (Gio)
p3.2xlarge	1	16	8	61
p3.8xlarge	4	64	32	244
p3.16xlarge	8	128	64	488
p3dn.24xlarge	8	256	96	768
p4d.24xlarge	8	320	96	1 152
p5.48xlarge	8	640	192	2048
g3s.xlarge	1	8	4	30,5
g3.4xlarge	1	8	16	122
g3.8xlarge	2	16	32	244
g3.16xlarge	4	32	64	488
g4dn.xlarge	1	16	4	16
g4dn.2xlarge	1	16	8	32
g4dn.4xlarge	1	16	16	64
g4dn.8xlarge	1	16	32	128

Type d'instance	GPU	Mémoire GPU (Gio)	vCPU	Mémoire (Gio)
g4dn.12xlarge	4	64	48	192
g4dn.16xlarge	1	16	64	256
g5.xlarge	1	24	4	16
g5.2xlarge	1	24	8	32
g5.4xlarge	1	24	16	64
g5.8xlarge	1	24	32	128
g5.16xlarge	1	24	64	256
g5.12xlarge	4	96	48	192
g5.24xlarge	4	96	96	384
g5.48xlarge	8	192	192	768
g6.xlarge	1	24	4	16
g6.2xlarge	1	24	8	32
g6.4xlarge	1	24	16	64
g6.8xlarge	1	24	32	128
g6.16.xlarge	1	24	64	256
g6.12xlarge	4	96	48	192
g6.24xlarge	4	96	96	384
g6.48xlarge	8	192	192	768
g6.metal	8	192	192	768
gr6.4xlarge	1	24	16	128

Type d'instance	GPU	Mémoire GPU (Gio)	vCPU	Mémoire (Gio)
g6e.xlarge	1	48	4	32
g6e.2xlarge	1	48	8	64
g6e.4xlarge	1	48	16	128
g6e.8xlarge	1	48	32	256
g6e16.xlarge	1	48	64	512
g6e12.xlarge	4	192	48	384
g6e24.xlarge	4	192	96	768
g6e48.xlarge	8	384	192	1536
gr6.8xlarge	1	24	32	256
g6f.large	1/8 (0,125)	3	2	8
g6f.xlarge	1/8 (0,125)	3	4	16
g6f.2xlarge	1/4 (0,25)	6	8	32
g6f.4xlarge	1/2 (0,5)	12	16	64
gr6f.4xlarge	1/2 (0,5)	12	16	128

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour Amazon ECS-optimized AMI en interrogeant le AWS Systems Manager API de stockage des paramètres. Avec ce paramètre, vous n'avez pas besoin de rechercher manuellement Amazon ECS-optimized Identifiants AMI. Pour plus d'informations sur l'API Systems Manager Parameter Store, voir GetParameter. L'utilisateur que vous utilisez doit disposer du `ssm:GetParameter` Autorisation IAM pour récupérer l'Amazon ECS-optimized Métadonnées AMI.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

Utilisation des GPU avec les instances gérées Amazon ECS

Les instances gérées Amazon ECS sont prises en charge GPU-accelerated informatique pour les charges de travail telles que l'apprentissage automatique, le calcul haute performance et le traitement vidéo via les types d'instances Amazon EC2 suivants. Pour plus d'informations sur les types d'instances pris en charge sur les instances gérées Amazon ECS, consultez la section Types d'instances gérées Amazon ECS.

Ce qui suit est un sous-ensemble de GPU-based types d'instances pris en charge sur les instances gérées Amazon ECS :

- `g4dn` : optimisé par NVIDIA T4 GPUs, adapté à l'inférence de machine learning, à la vision par ordinateur et aux applications gourmandes en graphismes.
- `g5` : optimisé par NVIDIA A10G GPUs, offrant des performances supérieures pour les applications à forte intensité graphique et les charges de travail de machine learning.
- `p3` : optimisé par NVIDIA V100 GPUs, conçu pour le calcul haute performance et l'entraînement du deep learning.
- `p4d` : optimisé par NVIDIA A100 GPUs, offrant les meilleures performances pour l'entraînement du machine learning et le calcul haute performance.
- `g6f`: Propulsé par NVIDIA L4 GPUs (fractionnel), fournissant des tranches de GPU partitionnées par matériel (1/8, 1/4, ou 1/2 d'un GPU NVIDIA L4) avec mémoire et calcul dédiés.

Quand vous utilisez GPU-enabled types d'instances avec Amazon ECS Managed Instances, le NVIDIA conducteurs et CUDA la boîte à outils est préinstallée sur l'instance, ce qui facilite son exécution GPU-accelerated charges de travail.

Vous pouvez également partager un seul GPU sur plusieurs conteneurs. Pour plus d'informations sur le partage de processeurs graphiques, consultez Partagez des GPU sur Amazon ECS.

GPU-enabled sélection d'instance

Pour sélectionner GPU-enabled types d'instances pour vos charges de travail Amazon ECS Managed Instances, utilisez `instanceRequirements` objet dans le modèle de lancement du fournisseur de capacité. L'extrait suivant montre les attributs qui peuvent être utilisés pour sélectionner GPU-enabled instances.

```
{
```

```
  "instanceRequirements": {
    "acceleratorTypes": "gpu",
    "acceleratorCount": 1,
    "acceleratorManufacturers": ["nvidia"]
  }
}
```

L'extrait suivant montre les attributs qui peuvent être utilisés pour spécifier GPU-enabled types d'instances dans le modèle de lancement.

```
{
  "instanceRequirements": {
    "allowedInstanceTypes": ["g4dn.xlarge", "p4de.24xlarge"]
  }
}
```

GPU-enabled images de conteneurs

Pour utiliser des GPU dans vos conteneurs, vous devez utiliser des images de conteneur contenant les bibliothèques et outils GPU nécessaires. NVIDIA fournit plusieurs images de conteneur prédéfinies que vous pouvez utiliser comme base pour vos charges de travail GPU, notamment les suivantes :

- `nvidia:cuda` : images de base avec le kit d'outils CUDA pour le calcul GPU.
- `tensorflow/tensorflow:latest-gpu` : TensorFlow avec support GPU.
- `pytorch/pytorch:latest-cuda` : PyTorch avec support GPU.

Pour un exemple de définition de tâche pour Amazon ECS sur instances gérées Amazon ECS impliquant l'utilisation de GPU, consultez la section Spécification des GPU dans une définition de tâche Amazon ECS.

Considérations

> **Note**
>
> La prise en charge du type de famille d'instances g2 est obsolète.
> Le type de famille d'instances p2 n'est pris en charge que sur les versions antérieures à `20230912` de l'Amazon ECS GPU-optimized AMI. Si vous devez continuer à utiliser des instances p2, veuillez consulter Que faire si vous avez besoin d'une instance P2.

In-place mises à jour du NVIDIA/CUDA les pilotes de ces deux types de familles d'instances peuvent entraîner des défaillances de la charge de travail du GPU.

Nous vous recommandons de prendre en compte ce qui suit avant de commencer à utiliser des GPU sur Amazon ECS.

- Vos clusters peuvent contenir une combinaison d'instances de conteneur GPU et non GPU.
- Vous pouvez exécuter des charges de travail GPU sur des instances externes. Lorsque vous enregistrez une instance externe auprès de votre cluster, assurez-vous que l'indicateur `--enable-gpu` est inclus dans le script d'installation. Pour plus d'informations, consultez Enregistrement d'une instance externe dans un cluster Amazon ECS.
- Vous devez définir `ECS_ENABLE_GPU_SUPPORT` sur `true` dans votre fichier de configuration d'agent. Pour plus d'informations, consultez the section called "Configuration de l'agent de conteneur".
- Lorsque vous exécutez une tâche ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches afin de déterminer sur quelles instances de conteneur la tâche est lancée. Ainsi, vous pouvez utiliser plus efficacement vos ressources. Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

  L'exemple suivant lance une tâche sur une instance de conteneur `g4dn.xlarge` dans votre cluster par défaut.

  ```
  aws ecs run-task --cluster default --task-definition ecs-gpu-task-def \
       --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
    g4dn.xlarge" --region us-east-2
  ```

- Pour chaque conteneur dont les besoins en ressources GPU sont spécifiés dans la définition du conteneur, Amazon ECS définit l'exécution du conteneur comme étant celle du conteneur NVIDIA.
- L'exécution du conteneur NVIDIA nécessite que certaines variables d'environnement soient définies dans le conteneur pour fonctionner correctement. Pour obtenir la liste de ces variables d'environnement, consultez la section Configurations spécialisées avec Docker. Amazon ECS définit la valeur de variable d'environnement `NVIDIA_VISIBLE_DEVICES` en tant que liste des ID de périphérique GPU assignés au conteneur par Amazon ECS. Pour les autres variables d'environnement requises, Amazon ECS ne les définit pas. Assurez-vous que votre image de conteneur les définit ou qu'elles sont définies dans la définition du conteneur.

- La famille de types d'instance p5 est prise en charge sur la version `20230929` et versions ultérieures d'Amazon ECS GPU-optimized AMI.
- La famille de types d'instance g4 est prise en charge sur la version `20230913` et versions ultérieures d'Amazon ECS GPU-optimized AMI. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux. Elle n'est pas prise en charge dans le flux de travail Create Cluster (Créer un cluster) dans la console Amazon ECS. Pour utiliser ces types d'instances, vous devez soit utiliser la console Amazon EC2, AWS CLI, ou API et enregistrez manuellement les instances dans votre cluster.
- Le type d'instance p4d.24xlarge ne fonctionne qu'avec CUDA 11 ou version ultérieure.
- L'Amazon ECS GPU-optimized IPv6 est activé pour AMI, ce qui pose des problèmes lors de l'utilisation `yum`. Cela peut être résolu en configurant `yum` pour utiliser IPv4 avec la commande suivante :

  ```
  echo "ip_resolve=4" >> /etc/yum.conf
  ```

- Lorsque vous créez une image de conteneur qui n'utilise pas le NVIDIA/CUDA images de base, vous devez définir `NVIDIA_DRIVER_CAPABILITIES` variable d'exécution du conteneur à l'une des valeurs suivantes :
  - `utility,compute`
  - `all`

  Pour savoir comment définir la variable, veuillez consulter Contrôle de l'exécution du conteneur NVIDIA sur le site Web de NVIDIA.
- Les GPU ne sont pas pris en charge sur les conteneurs Windows.

Lancement d'une instance de conteneur GPU pour Amazon ECS

Pour utiliser une instance GPU sur Amazon ECS avec Amazon EC2, vous devez créer un modèle de lancement, un fichier de données utilisateur, puis lancer l'instance.

Vous pouvez ensuite exécuter une tâche qui utilise une définition de tâche configurée pour le GPU.

Utiliser un modèle de lancement

Vous pouvez créer un modèle de lancement.

- Créez un modèle de lancement qui utilise Amazon ECS-optimized ID d'AMI GPU pour l'AMI. Pour plus d'informations sur la création d'un modèle de lancement, consultez la section Créer un

modèle de lancement à l'aide des paramètres que vous définissez dans le Guide de l'utilisateur Amazon EC2.

Utilisez l'ID d'AMI de l'étape précédente pour l'Amazon Machine image. Pour plus d'informations sur la façon de spécifier l'ID AMI avec le paramètre Systems Manager, consultez la section Spécifier un paramètre Systems Manager dans un modèle de lancement dans le Guide de l'utilisateur Amazon EC2.

Ajoutez ce qui suit à la section Données utilisateur du modèle de lancement. Remplacez *cluster-name* par le nom de votre cluster.

```
#!/bin/bash
echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
```

Utilisez la commande AWS CLI

Vous pouvez utiliser le AWS CLI pour lancer l'instance de conteneur.

1. Créez un fichier appelé `userdata.toml`. Ce fichier est utilisé pour les données utilisateur de l'instance. Remplacez *cluster-name* par le nom de votre cluster.

   ```
   #!/bin/bash
   echo ECS_CLUSTER=cluster-name >> /etc/ecs/ecs.config;
   echo ECS_ENABLE_GPU_SUPPORT=true >> /etc/ecs/ecs.config
   ```

2. Exécutez la commande suivante pour obtenir l'ID d'AMI GPU. Utilisez ceci lors de l'étape suivante.

   ```
   aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
   recommended --region us-east-1
   ```

3. Exécutez la commande suivante pour lancer l'instance GPU. N'oubliez pas de remplacer les paramètres suivants :

   - Remplacer *subnet* avec l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - Remplacer *gpu_ami* à l'aide de l'ID AMI de l'étape précédente.
   - Remplacer *t3.large* avec le type d'instance que vous souhaitez utiliser.

- Remplacer *region* avec le code de région.

```
aws ec2 run-instances --key-name ecs-gpu-example \
   --subnet-id subnet \
   --image-id gpu_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications 'ResourceType=instance,Tags=[{Key=GPU,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Exécutez la commande suivante pour vérifier que l'instance de conteneur est enregistrée auprès du cluster. Lorsque vous exécutez cette commande, n'oubliez pas de remplacer les paramètres suivants :

   - Remplacez *cluster* par le nom de votre cluster.
   - Remplacer *region* à l'aide de votre code de région.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Spécification des GPU dans une définition de tâche Amazon ECS

Pour utiliser plusieurs GPU sur une instance de conteneur et l'exécution du GPU de Docker, assurez-vous de désigner le nombre de GPU nécessaires à votre conteneur dans la définition de tâche. Vous pouvez spécifier une valeur entière, une valeur décimale pour les GPU fractionnaires ou `ALL`. Lorsque vous spécifiez `ALL`, tous les GPU de l'instance de conteneur sont alloués au conteneur. Lorsque vous spécifiez une valeur décimale telle que `0.25`, Amazon ECS place la tâche sur une instance GPU fractionnée qui fournit la capacité GPU requise. Une fois les conteneurs supportant les GPU placés, l'agent de conteneur Amazon ECS épinglera le nombre souhaité de GPU physiques au bon conteneur. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne peut pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Pour partager un seul GPU physique entre plusieurs charges de travail au lieu d'allouer un GPU entier ou fractionné à chaque conteneur, voir Partagez des GPU sur Amazon ECS.

> **Important**
>
> Si vos exigences en matière de GPU ne sont pas spécifiées dans la définition de tâche, la tâche utilise l'exécution par défaut du Docker.

L'exemple suivant illustre le format JSON pour les exigences de GPU dans une définition de tâche :

```
{
  "containerDefinitions": [
     {
        ...
        "resourceRequirements" : [
            {
               "type" : "GPU",
               "value" : "2"
            }
        ],
     },
...
}
```

Vous pouvez également spécifier `ALL` comme valeur au lieu d'un nombre pour allouer tous les GPU de l'instance de conteneur au conteneur.

L'exemple suivant illustre la syntaxe d'un conteneur Docker qui spécifie une exigence GPU. Ce conteneur utilise 2 GPU, exécute l'utilitaire `nvidia-smi`, puis se ferme.

```
{
  "containerDefinitions": [
    {
      "memory": 80,
      "essential": true,
      "name": "gpu",
      "image": "nvidia/cuda:11.0.3-base",
      "resourceRequirements": [
         {
           "type":"GPU",
           "value": "2"
         }
      ],
      "command": [
```

```
        "sh",
        "-c",
        "nvidia-smi"
      ],
      "cpu": 100
    }
  ],
  "family": "example-ecs-gpu"
}
```

L'exemple de définition de tâche suivant montre un conteneur TensorFlow qui affiche le nombre de GPU disponibles. La tâche s'exécute sur des instances gérées Amazon ECS, nécessite un GPU et utilise une instance `g4dn.xlarge`.

```
{
  "family": "tensorflow-gpu",
  "networkMode": "awsvpc",
  "executionRoleArn": "arn:aws:iam::account-id:role/ecsTaskExecutionRole",
  "containerDefinitions": [
    {
      "name": "tensorflow",
      "image": "tensorflow/tensorflow:latest-gpu",
      "essential": true,
      "command": [
        "python",
        "-c",
        "import tensorflow as tf; print('Num GPUs Available: ',
 len(tf.config.list_physical_devices('GPU')))"
      ],
      "resourceRequirements": [
        {
          "type": "GPU",
          "value": "1"
        }
      ],
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/tensorflow-gpu",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "ecs"
        }
      }
```

```
    }
  ],
  "requiresCompatibilities": [
    "MANAGED_INSTANCES"
  ],
  "cpu": "4096",
  "memory": "8192",
}
```

Spécification de GPU fractionnaires

Amazon ECS prend en charge la planification fractionnée des GPU avec les instances Amazon EC2 G6f. Les instances G6f fournissent des fractions partitionnées matériellement d'un NVIDIA L4 GPU, où chaque instance expose une tranche de GPU fixe (1/8, 1/4, ou 1/2) avec mémoire GPU et calcul dédiés. Pour utiliser un GPU fractionnaire, spécifiez une valeur décimale dans la définition de votre conteneur `resourceRequirements`:

```
"resourceRequirements": [
  {
    "type": "GPU",
    "value": "0.25"
  }
]
```

Les valeurs fractionnaires prises en charge et les types d'instances G6f correspondants sont les suivants :

Valeur du GPU	Fraction GPU	Mémoire GPU	Types d’instances
`0.125`	1/8 GPU	3 Go	g6f.large, g6f.xlarge
`0.25`	1/4 GPU	6 GO	g6f.2xlarge
`0.5`	1/2 GPU	12 GO	g 6 f x 4 x large, g 6 f x 4 x grand

L'exemple de définition de tâche suivant exécute un conteneur d'inférence léger sur un GPU fractionnaire avec 1/4 d'un NVIDIA L4 GPU (6 Go de mémoire GPU) :

```
{
```

```
  "family": "fractional-gpu-inference",
  "networkMode": "awsvpc",
  "executionRoleArn": "arn:aws:iam::account-id:role/ecsTaskExecutionRole",
  "containerDefinitions": [
    {
      "name": "inference",
      "image": "nvidia/cuda:12.0.0-base-ubuntu22.04",
      "essential": true,
      "command": ["nvidia-smi"],
      "resourceRequirements": [
        {
          "type": "GPU",
          "value": "0.25"
        }
      ],
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fractional-gpu-inference",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "ecs"
        }
      }
    }
  ],
  "requiresCompatibilities": [
    "MANAGED_INSTANCES"
  ],
  "cpu": "2048",
  "memory": "4096"
}
```

Considérations relatives au GPU fractionnel

- Un seul conteneur par tâche peut demander une valeur GPU fractionnaire. Si un conteneur spécifie un GPU fractionnaire, aucun autre conteneur de cette définition de tâche ne peut avoir besoin de ressources GPU.
- Valeurs entières (`1`, `2`, etc.) sont traitées comme des requêtes GPU complètes et ne sont placées que sur des instances dotées d'une capacité GPU complète. Valeurs décimales (`0.125`, `0.25`, `0.5`) sont traitées comme des requêtes GPU fractionnaires et placées sur des instances avec la partition GPU fractionnée correspondante.

- Vous pouvez inclure à la fois des instances GPU fractionnées (G6f) et des instances GPU complètes (telles que g5 ou g6) dans le même fournisseur de capacité. Amazon ECS achemine les tâches vers le type d'instance approprié en fonction de la valeur GPU déclarée. Vous n'avez pas besoin de fournisseurs de capacité distincts par famille de GPU.
- Vous pouvez utiliser la planification fractionnée du GPU avec Amazon ECS Managed Instances et Amazon ECS sur Amazon EC2. Fargate et Amazon ECS Anywhere ne prennent pas en charge la planification fractionnée des GPU.
- Pour les charges de travail GPU complètes, continuez à utiliser des valeurs entières comme précédemment : `"value": "1"`.

Partagez des GPU sur Amazon ECS

De nombreuses charges de travail GPU, telles que l'inférence et la diffusion de petits modèles, n'utilisent qu'une fraction du calcul et de la mémoire d'un GPU. Sans partage, le reste de cette capacité reste inactif. Avec Amazon ECS, vous pouvez partager un seul document physique NVIDIA GPU sur plusieurs conteneurs, ce qui vous permet de regrouper davantage de charges de travail sur le même matériel et de réduire le gaspillage de GPU.

Avec Amazon ECS, vous pouvez utiliser NVIDIA Multi-Process Service (MPS) pour le partage de GPU. MPS est un NVIDIA CUDA fonctionnalité qui exécute plusieurs processus simultanément sur le même GPU, ce qui améliore l'utilisation du GPU et réduit les frais liés au changement de contexte. MPS fournit un partitionnement de la mémoire et du calcul, de sorte que chaque conteneur bénéficie d'une limite de mémoire configurable et d'un plafond de calcul optionnel.

Amazon ECS gère MPS pour toi. Il démarre le MPS démon sur les instances GPU, place les conteneurs sur les GPU en fonction de la mémoire disponible et applique des limites par conteneur.

MPS fournit un partage au niveau logiciel plutôt qu'un partitionnement au niveau matériel. Pour plus d'informations sur MPS, voir Multi-Process Service dans le NVIDIA documentation.

Amazon ECS prend en charge le partage de GPU avec MPS sur les instances gérées Amazon ECS et Amazon ECS sur Amazon EC2.

Quand utiliser le partage GPU

Partage de GPU avec MPS convient parfaitement pour :

- Charges de travail d'inférence, génération d'intégration et serveurs de petits modèles qui n'utilisent pas pleinement un GPU

- Multi-model serveur où plusieurs modèles peuvent partager un seul GPU
- les environnements de développement et de test,
- Exécution de plusieurs charges de travail GPU légères sur le même matériel pour réduire les coûts

Le partage de GPU n'est pas recommandé pour :

- Entraînement sur grands modèles nécessitant toute la mémoire du GPU
- Charges de travail sensibles aux défaillances du processeur graphique provenant de conteneurs voisins, car MPS est un partage au niveau logiciel et tous les conteneurs d'un GPU partagent la même chose CUDA contexte
- Charges de travail nécessitant une isolation matérielle entre les processus

Comment fonctionne le partage de GPU

Vous configurez le partage GPU dans la définition du conteneur d'une définition de tâche. Dans chaque contenant `resourceRequirements`, vous fournissez un `sharingStrategy` avec une valeur de mémoire GPU au lieu de spécifier un entier dans `value` pour un GPU dédié.

Lorsqu'Amazon ECS place une tâche qui utilise le partage GPU, il additionne la mémoire GPU demandée par tous les conteneurs GPU de la tâche et recherche un GPU disposant de suffisamment de mémoire libre. Tous les conteneurs GPU d'une même tâche sont placés sur le même GPU physique. Si la tâche demande 4 000 Mo sur deux conteneurs, Amazon ECS trouve un seul GPU avec au moins 4 000 Mo de mémoire restante et y place les deux conteneurs.

Plusieurs tâches peuvent également partager le même GPU physique. Tant qu'il reste suffisamment de mémoire sur un GPU, Amazon ECS peut lui confier des tâches supplémentaires, même si ces tâches proviennent de différents services. Une tâche qui partage un GPU et une tâche qui utilise un GPU dédié peuvent être exécutées sur la même instance, sur des GPU différents.

Sur les instances gérées Amazon ECS, Amazon ECS surveille l'état des GPU et MPS démon sur chaque instance. Si Amazon ECS détecte une défaillance matérielle du processeur graphique ou MPS en cas de défaillance du démon, il vide l'instance et la remplace. Pour plus d'informations sur la réparation automatique des GPU, consultez Réparation automatique du processeur graphique pour les instances gérées par Amazon ECS.

Configurer le partage du GPU

Pour partager un GPU, ajoutez `sharingStrategy` s'opposer à `resourceRequirements` de chaque conteneur GPU dans votre définition de tâche. Le `sharingStrategy` l'objet contient un `nvidiaMps` objet avec les champs suivants. Pour plus d'informations sur `resourceRequirements`, consultez Besoins en ressources.

`memory`

Obligatoire : oui

Limite de mémoire GPU pour le conteneur, en MiB. Cette valeur représente un plafond strict de la mémoire GPU que le conteneur peut utiliser. La valeur doit être supérieure à `0`.

Pour connaître la mémoire GPU totale disponible sur une instance, utilisez le `DescribeContainerInstances` API (la `totalMemoryInMiB` champ) ou vérifiez les spécifications du GPU pour votre type d'instance.

`maxComputePercent`

Obligatoire : non

Le pourcentage maximum de calcul GPU que le conteneur peut utiliser, sous la forme d'un entier de 1 pour `100`. Cela limite l'utilisation du processeur graphique du conteneur, et non une réservation. Lorsqu'un conteneur est inactif, son calcul inutilisé est disponible pour d'autres conteneurs sur le même GPU, chacun d'entre eux pouvant utiliser le sien `maxComputePercent` valeurs. Si vous ne spécifiez pas ce paramètre, le conteneur peut utiliser jusqu'à `100` pour cent.

Amazon ECS place les tâches uniquement en fonction de la mémoire du GPU, de sorte qu'il ne prend pas en compte le calcul lors de la prise de décisions de placement.

Vous pouvez définir la stratégie de partage dans l'éditeur de définition des tâches de la console Amazon ECS, ou avec JSON, AWS CLI, ou l'API.

Amazon ECS définit les variables d'environnement réservées suivantes dans chaque conteneur qui partage un GPU. Ne définissez pas ces variables dans la définition de votre conteneur.

- `CUDA_MPS_PINNED_DEVICE_MEM_LIMIT` pour la limite de mémoire
- `CUDA_MPS_ACTIVE_THREAD_PERCENTAGE` pour le plafond de calcul

L'exemple de définition de tâche suivant exécute deux conteneurs d'inférence qui partagent un GPU avec MPS, ainsi qu'un conteneur sidecar non GPU. Chaque conteneur GPU demande une quantité différente de mémoire GPU et de calcul.

Dans l'exemple suivant, remplacez *account-id* avec votre AWS ID de compte.

```
{
  "family": "mps-inference",
  "networkMode": "awsvpc",
  "executionRoleArn": "arn:aws:iam::account-id:role/ecsTaskExecutionRole",
  "containerDefinitions": [
    {
      "name": "model-a",
      "image": "nvidia/cuda:12.0.0-base-ubuntu22.04",
      "essential": true,
      "resourceRequirements": [
        {
          "type": "GPU",
          "sharingStrategy": {
            "nvidiaMps": {
              "memory": 12288,
              "maxComputePercent": 50
            }
          }
        }
      ]
    },
    {
      "name": "model-b",
      "image": "nvidia/cuda:12.0.0-base-ubuntu22.04",
      "essential": true,
      "resourceRequirements": [
        {
          "type": "GPU",
          "sharingStrategy": {
            "nvidiaMps": {
              "memory": 8192,
              "maxComputePercent": 25
            }
          }
        }
      ]
    },
```

```
    {
      "name": "log-router",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:stable",
      "essential": false
    }
  ],
  "requiresCompatibilities": [
    "MANAGED_INSTANCES"
  ],
  "cpu": "4096",
  "memory": "8192"
}
```

Règles de validation

Amazon ECS applique les règles suivantes aux configurations de partage de GPU :

- Un besoin en ressources GPU utilise soit `value` (un GPU dédié) ou `sharingStrategy` (un GPU partagé), mais pas les deux. Lorsque vous spécifiez `sharingStrategy`, vous devez omettre `value`.
- Tous les conteneurs GPU d'une tâche doivent utiliser la même stratégie de partage. Si un contenant utilise `sharingStrategy`, aucun autre conteneur de cette tâche ne peut demander un GPU dédié avec `value`.
- Le `memory` la valeur est requise et doit être supérieure à `0`.
- La valeur `maxComputePercent` est facultative. Si vous le spécifiez, la valeur doit être un entier compris entre 1 pour `100`.
- Pour les instances gérées Amazon ECS, la tâche `requiresCompatibilities` doit inclure `MANAGED_INSTANCES`. Pour Amazon ECS sur Amazon EC2, utilisez `EC2` type de lancement.
- Le partage GPU n'est pas pris en charge pour les définitions de tâches qui nécessitent `EXTERNAL` compatibilité (Amazon ECS Anywhere).
- Vous ne pouvez pas spécifier `sharingStrategy` dans une dérogation de conteneur. Dérogations de conteneurs dans le `RunTask` et `StartTask` les requêtes ne prennent pas en charge la configuration de partage du GPU, alors définissez `sharingStrategy` dans la définition de la tâche à la place. Les services exécutent la définition de tâche telle qu'elle est enregistrée et ne sont pas affectés.

Configurer le partage GPU par type de lancement

Avec Amazon ECS Managed Instances, Amazon ECS sélectionne et gère les instances GPU pour vous. Vous n'avez besoin d'aucune configuration au-delà de la définition de la tâche.

Avec Amazon ECS sur Amazon EC2, lancez un NVIDIA Instance GPU avec Amazon ECS AL2023 GPU-optimized AMI. Partage de GPU avec MPS sur Amazon ECS sur Amazon EC2 nécessite la version agent de conteneur Amazon ECS `1.107.0` ou version ultérieure et Amazon ECS AL2023 GPU-optimized Version AMI `20261019` ou plus tard. Une fois l'instance enregistrée auprès de votre cluster, la configuration de définition des tâches est la même que pour les instances gérées Amazon ECS.

Considérations

- Partage de GPU avec MPS est compatible avec les modèles G4dn, G5, G6, G6e, G7, G7e, P3, P4d, P5, P5e, P5en et P6-B200 familles d'instances. Le partage de GPU n'est pas pris en charge sur AWS Fargate, Amazon ECS Express Mode ou tâches de démon gérées par Amazon ECS.
- Partage de GPU avec MPS sur Amazon ECS sur Amazon EC2 nécessite la version agent de conteneur Amazon ECS `1.107.0` ou version ultérieure et Amazon ECS AL2023 GPU-optimized Version AMI `20261019` ou plus tard.
- Exécutez un seul processus GPU par conteneur lorsque vous utilisez NVIDIA MPS pour le partage de GPU. Pour les applications qui nécessitent plusieurs processus GPU, configurez la mémoire GPU requise en fonction de l'utilisation maximale combinée de la mémoire de tous les processus. Utilisez les contrôles au niveau de l'application pour maintenir la consommation totale dans les limites de la valeur configurée. Amazon ECS utilise cette valeur pour déterminer l'emplacement des conteneurs sur les GPU partagés. Par conséquent, les applications multiprocessus peuvent consommer plus de mémoire GPU que ce qui est configuré, ce qui peut entraîner des problèmes de mémoire.
- Si un conteneur déclenche une erreur GPU fatale, telle qu'un accès illégal à la mémoire depuis un noyau GPU défectueux, les conteneurs partageant ce GPU physique peuvent être affectés. Les conteneurs des autres GPU de la même instance ne sont pas affectés.
- Le `maxComputePercent` La valeur est un plafond et non une réservation garantie. Lorsqu'un conteneur est inactif, son calcul inutilisé est disponible pour d'autres conteneurs sur le même GPU, chacun d'entre eux pouvant utiliser le sien `maxComputePercent` valeurs.
- Le partage GPU n'est pas pris en charge pour les définitions de tâches qui nécessitent `EXTERNAL` compatibilité (Amazon ECS Anywhere).

- Les tâches provenant de différents services utilisant le même fournisseur de capacité peuvent être placées sur le même GPU. Pour empêcher les tâches de différents services de partager un GPU, utilisez des fournisseurs de capacité distincts ou des clusters distincts pour ces services.
- Pour les GPU partagés avec MPS, Amazon ECS n'émet pas les métriques GPU suivantes par tâche et par conteneur :
  - Le `TaskGPUUtilization` et `ContainerGPUUtilization` métriques.
  - Le `TaskGPUMemoryUtilization` et `ContainerGPUMemoryUtilization` métriques.
  - Le `TaskGPUMemoryTotal` et `ContainerGPUMemoryTotal` métriques.
  - Le `TaskGPUMemoryUsed` et `ContainerGPUMemoryUsed` métriques.
- Les métriques GPU par tâche et par conteneur suivantes restent disponibles pour les GPU partagés avec MPS:
  - Le `TaskGPUPowerDraw` et `ContainerGPUPowerDraw` métriques.
  - Le `TaskGPUTemperature` et `ContainerGPUTemperature` métriques.
  - Le `TaskGPURestartAppXidCount` et `ContainerGPURestartAppXidCount` métriques.
- Instance-level Les métriques du GPU restent disponibles, notamment `InstanceGPUUsageTotal` et `InstanceGPULimit`.

Surveillez le partage du GPU

Vous pouvez confirmer qu'un GPU est partagé et suivre son utilisation à l'aide de l'API Amazon ECS et d'Amazon CloudWatch métriques.

Confirmez qu'un GPU est partagé

Pour confirmer qu'un GPU est partagé, appelez le `DescribeContainerInstances` API. La réponse inclut `gpuDevices` champ avec le mode de partage, le nombre de conteneurs et la mémoire restante pour chaque GPU. Voici un exemple de réponse.

```
{
  "containerInstances": [
    {
      "ec2InstanceId": "i-0c74b75400ab4228d",
      "runningTasksCount": 1,
      "gpuDevices": [
        {
          "gpuId": "GPU-70d3bdaf-8635-81e0-8e44-446248a92c83",
          "sharingMode": "NVIDIA_MPS",
```

```
          "totalMemoryInMiB": 22587,
          "remainingMemoryInMiB": 16587,
          "containerCount": 2
        }
      ]
    }
  ]
}
```

Afficher la configuration par tâche

Pour afficher la configuration de partage pour chaque tâche, appelez le `DescribeTasks` API. Chaque contenant `gpuSharingStrategy` affiche ses limites de mémoire et de calcul, et les conteneurs qui partagent un GPU répertorient la même valeur dans `gpuIds`. Voici un exemple de réponse.

```
{
  "tasks": [
    {
      "lastStatus": "RUNNING",
      "launchType": "MANAGED_INSTANCES",
      "containers": [
        {
          "name": "mps-a",
          "gpuIds": ["GPU-70d3bdaf-8635-81e0-8e44-446248a92c83"],
          "gpuSharingStrategy": { "nvidiaMps": { "memory": 4000, "maxComputePercent":
 50 } }
        },
        {
          "name": "mps-b",
          "gpuIds": ["GPU-70d3bdaf-8635-81e0-8e44-446248a92c83"],
          "gpuSharingStrategy": { "nvidiaMps": { "memory": 2000, "maxComputePercent":
 50 } }
        }
      ]
    }
  ]
}
```

Suivez l'utilisation du GPU

- Densité de tâches plus élevée: avec le partage de GPU, davantage de tâches sont exécutées sur la même instance. Utilisez le `RunningTaskCount` métrique permettant de comparer le nombre de tâches par service avant et après l'activation du partage. Pour la densité de tâches par instance, utilisez le `instance_number_of_running_tasks` métrique, qui CloudWatch Container Insights collecte pour chaque instance EC2 lorsque vous déployez CloudWatch agent pour collecter des métriques au niveau de l'instance EC2.
- Moins de GPU pour une charge de travail identique: Les `InstanceGPUUsageTotal` métrique en CloudWatch Container Insights indique le nombre de GPU physiques utilisés. Avec MPS, la même charge de travail utilise moins de GPU.
- Hauteur du processeur graphique de l'instance: Comparez `InstanceGPUUsageTotal` avec `InstanceGPULimit` pour voir le niveau de remplissage d'une instance. Avec MPS, davantage de tâches peuvent être effectuées avant qu'une instance ne soit à court de GPU.

Partagez des GPU manuellement sur Amazon ECS sur Amazon EC2

Pour partager des GPU manuellement sur Amazon ECS sur Amazon EC2, vous configurez et gérez vous-même le partage des GPU. Utilisez plutôt l'approche gérée, dans laquelle Amazon ECS configure et gère le partage pour vous. Pour plus d'informations sur l'approche gérée, voir Partagez des GPU sur Amazon ECS.

Lorsque vous souhaitez partager des GPU, vous devez configurer les éléments suivants.

1. Supprimez les exigences en matière de ressources GPU de vos définitions de tâches afin qu'Amazon ECS ne réserve aucun GPU à partager.
2. Ajoutez les données utilisateur suivantes à vos instances lorsque vous souhaitez partager des GPU. NVIDIA deviendra ainsi le moteur d'exécution par défaut du conteneur Docker sur l'instance de conteneur afin que tous les conteneurs Amazon ECS puissent utiliser les GPU. Pour plus d'informations, consultez la section Exécution de commandes lors du lancement d'une instance EC2 avec des données utilisateur dans le Guide de l'utilisateur Amazon EC2.

```
const userData = ec2.UserData.forLinux();
 userData.addCommands(
 'sudo rm /etc/sysconfig/docker',
 'echo DAEMON_MAXFILES=1048576 | sudo tee -a /etc/sysconfig/docker',
 'echo OPTIONS="--default-ulimit nofile=32768:65536 --default-runtime nvidia" |
 sudo tee -a /etc/sysconfig/docker',
```

```
 'echo DAEMON_PIDFILE_TIMEOUT=10 | sudo tee -a /etc/sysconfig/docker',
 'sudo systemctl restart docker',
);
```

3. Définissez la variable d'environnement `NVIDIA_VISIBLE_DEVICES` sur votre conteneur. Pour ce faire, vous pouvez spécifier la variable d'environnement dans votre définition de tâche. Pour plus d'informations sur les valeurs valides, consultez la sectionÉnumération des GPU sur le site de documentation de NVIDIA.

Que faire si vous avez besoin d'une instance P2

Si vous avez besoin d'utiliser une instance P2, vous pouvez utiliser l'une des options suivantes pour continuer à utiliser ces instances.

Vous devez modifier les données utilisateur de l'instance pour les deux options. Pour plus d'informations, consultez la section Exécution de commandes lors du lancement d'une instance EC2 avec des données utilisateur dans le Guide de l'utilisateur Amazon EC2.

Utilisez la dernière prise en charge GPU-optimized AMI

Vous pouvez utiliser le `20230906` version du GPU-optimized AMI, et ajoutez ce qui suit aux données utilisateur de l'instance.

Remplacez cluster-name par le nom de votre cluster.

```
#!/bin/bash
echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
```

Utilisez les dernières GPU-optimized AMI, et mettez à jour les données utilisateur

Vous pouvez ajouter ce qui suit aux données utilisateur de l'instance. Cela désinstalle le Nvidia 535/Cuda12.2 pilotes, puis installe le Nvidia 470/Cuda114 pilotes et corrige la version.

```
#!/bin/bash
yum remove -y cuda-toolkit* nvidia-driver-latest-dkms*
tmpfile=$(mktemp)
cat >$tmpfile <<EOF
[amzn2-nvidia]
name=Amazon Linux 2 Nvidia repository
mirrorlist=\$awsproto://\$amazonlinux.\$awsregion.\$awsdomain/\$releasever/amzn2-
nvidia/latest/\$basearch/mirror.list
```

```
priority=20
gpgcheck=1
gpgkey=https://developer.download.nvidia.com/compute/cuda/repos/rhel7/
x86_64/7fa2af80.pub
enabled=1
exclude=libglvnd-*
EOF

mv $tmpfile /etc/yum.repos.d/amzn2-nvidia-tmp.repo
yum install -y system-release-nvidia cuda-toolkit-11-4 nvidia-driver-latest-
dkms-470.182.03
yum install -y libnvidia-container-1.4.0 libnvidia-container-tools-1.4.0 nvidia-
container-runtime-hook-1.4.0 docker-runtime-nvidia-1

echo "exclude=*nvidia* *cuda*" >> /etc/yum.conf
nvidia-smi
```

Créez votre propre compatible P2 GPU-optimized AMI

Vous pouvez créer votre propre Amazon ECS personnalisé GPU-optimized AMI compatible avec les instances P2, puis lancez des instances P2 à l'aide de l'AMI.

1. Exécutez la commande suivante pour cloner le `amazon-ecs-ami repo`.

   ```
   git clone https://github.com/aws/amazon-ecs-ami
   ```

2. Définissez l'agent Amazon ECS requis et les versions de l'AMI Amazon Linux source dans `release.auto.pkrvars.hcl` ou `overrides.auto.pkrvars.hcl`.
3. Exécutez la commande suivante pour créer une AMI EC2 privée compatible avec P2.

   Remplacez la région par la région de l'instance.

   ```
   REGION=region make al2keplergpu
   ```

4. Utilisez l'AMI avec les données utilisateur de l'instance suivantes pour vous connecter au cluster Amazon ECS.

   Remplacez cluster-name par le nom de votre cluster.

   ```
   #!/bin/bash
   echo "ECS_CLUSTER=cluster-name" >> /etc/ecs/ecs.config
   ```

Définitions de tâche Amazon ECS pour les charges de travail de transcodage vidéo

Pour utiliser des charges de travail de transcodage vidéo sur Amazon ECS, enregistrez des instances Amazon EC2 VT1. Après avoir enregistré ces instances, vous pouvez exécuter des charges de travail de transcodage vidéo en direct et prérendues en tant que tâches sur Amazon ECS. Les instances Amazon EC2 VT1 utilisent des cartes de transcodage multimédia Xilinx U30 pour accélérer les charges de travail de transcodage vidéo en direct et prérendues.

> **Note**
>
> Pour obtenir des instructions sur l'exécution de charges de travail de transcodage vidéo dans des conteneurs autres qu'Amazon ECS, consultez la documentation Xilinx.

Considérations

Avant de commencer à déployer VT1 sur Amazon ECS, prenez en compte ce qui suit :

- Vos clusters peuvent contenir une combinaison d'instances VT1 et non VT1.
- Vous avez besoin d'une application Linux qui utilise des cartes de transcodage multimédia Xilinx U30 avec AVC accéléré (H.264) et HEVC (H.265) codecs.

  > **Important**
  >
  > Les applications qui utilisent d'autres codecs peuvent ne pas avoir de performances améliorées sur les instances VT1.

- Une seule tâche de transcodage peut être exécutée sur une carte U30. Chaque carte est associée à deux appareils. Vous pouvez exécuter autant de tâches de transcodage que de cartes pour chacune de vos instances VT1.
- Lorsque vous exécutez une tâche autonome ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches. Cela garantit que la tâche est lancée sur l'instance de conteneur que vous spécifiez. Cela permet de vous assurer que vous utilisez efficacement vos ressources et que vos tâches pour vos charges de travail de transcodage vidéo se trouvent sur vos instances VT1. Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

Dans l'exemple suivant, une tâche est exécutée sur une instance `vt1.3xlarge` de votre cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition vt1-3xlarge-xffmpeg-processor \
     --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
 vt1.3xlarge"
```

- Vous configurez un conteneur pour utiliser une carte U30 spécifique disponible sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération en utilisant le paramètre `linuxParameters` et en spécifiant les détails de l'appareil. Pour plus d'informations, consultez Exigences relatives à la définition de tâche.

Utilisation d'une AMI VT1

Vous avez deux options pour exécuter une AMI sur des instances de conteneur Amazon EC2 pour Amazon ECS. La première option consiste à utiliser l'AMI officielle Xilinx sur AWS Marketplace. La deuxième option consiste à créer votre propre AMI à partir de l'exemple du référentiel.

- Xilinx propose des AMI sur le AWS Marketplace.
- Amazon ECS fournit un exemple de référentiel que vous pouvez utiliser pour créer une AMI pour les charges de travail de transcodage vidéo. Cette AMI est fournie avec les pilotes Xilinx U30. Vous pouvez trouver le référentiel qui contient les scripts Packer sur GitHub. Pour plus d'informations sur Packer, consultez la documentation Packer.

Exigences relatives à la définition de tâche

Pour exécuter des conteneurs de transcodage vidéo sur Amazon ECS, votre définition de tâche doit contenir une application de transcodage vidéo qui utilise le H.264/AVC et H.265/HEVC codecs. Vous pouvez créer une image de conteneur en suivant les étapes du Xilinx GitHub.

La définition de tâche doit être spécifique au type d'instance. Les types d'instance sont 3xlarge, 6xlarge et 24xlarge. Vous devez configurer un conteneur pour utiliser des appareils Xilinx U30 spécifiques disponibles sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération à l'aide du paramètre `linuxParameters`. Le tableau suivant détaille les cartes et l'appareil SoCs qui sont spécifiques à chaque type d'instance.

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 adressables	Chemins de l'appareil
vt1.3xlarge	12	24	1	2	`/dev/dri/renderD128 ,/dev/dri/renderD129`
vt1.6xlarge	24	48	2	4	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131`
vt1.24xlarge	96	182	8	16	`/dev/dri/renderD128 ,/dev/dri/renderD129 ,/dev/dri/renderD130 ,/dev/dri/renderD131 ,/dev/dri/`

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 adressables	Chemins de l'appareil
					renderD132 ,/dev/dri/renderD133 ,/dev/dri/renderD134 ,/dev/dri/renderD135 ,/dev/dri/renderD136 ,/dev/dri/renderD137 ,/dev/dri/renderD138 ,/dev/dri/renderD139 ,/dev/dri/renderD140 ,/dev/dri/renderD141 ,/dev/dri/renderD142 ,/dev/

Type d'instance	vCPU	RAM (Gio)	Cartes accélérat rices U30	Appareils SoC XCU30 adressables	Chemins de l'appareil
					`dri/renderD143`

> **Important**
>
> Si la définition de tâche répertorie les périphériques dont l'instance EC2 ne dispose pas, la tâche ne s'exécute pas. Lorsque la tâche échoue, le message d'erreur suivant s'affiche dans `stoppedReason` : `CannotStartContainerError: Error response from daemon: error gathering device information while adding custom device "/dev/dri/renderD130": no such file or directory`.

Spécification du transcodage vidéo dans une définition de tâche Amazon ECS

Dans l'exemple suivant, la syntaxe utilisée pour définir une tâche d'un conteneur Linux sur Amazon EC2 est fournie. Cette définition de tâche concerne les images de conteneur créées conformément à la procédure fournie dans la documentation Xilinx. Si vous utilisez cet exemple, remplacez `image` par votre propre image, et copiez vos fichiers vidéo dans l'instance dans le répertoire `/home/ec2-user`.

vt1.3xlarge

1. Créez un fichier texte nommé `vt1-3xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-3xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
```

```
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.3xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
```

```
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-3xlarge-xffmpeg-processor --cli-
input-json file://vt1-3xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.6xlarge

1. Créez un fichier texte nommé `vt1-6xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-6xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.6xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
                "devices": [
                    {
```

```
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
```

```
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-6xlarge-xffmpeg-processor --cli-
input-json file://vt1-6xlarge-xffmpeg-linux.json --region us-east-1
```

vt1.24xlarge

1. Créez un fichier texte nommé `vt1-24xlarge-ffmpeg-linux.json` avec le contenu suivant.

```
{
    "family": "vt1-24xlarge-xffmpeg-processor",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == vt1.24xlarge"
        }
    ],
    "containerDefinitions": [
        {
            "entryPoint": [
                "/bin/bash",
                "-c"
            ],
            "command": ["/video/ecs_ffmpeg_wrapper.sh"],
            "linuxParameters": {
```

```
                "devices": [
                    {
                        "containerPath": "/dev/dri/renderD128",
                        "hostPath": "/dev/dri/renderD128",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD129",
                        "hostPath": "/dev/dri/renderD129",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD130",
                        "hostPath": "/dev/dri/renderD130",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD131",
                        "hostPath": "/dev/dri/renderD131",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD132",
                        "hostPath": "/dev/dri/renderD132",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD133",
                        "hostPath": "/dev/dri/renderD133",
```

```
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD134",
                        "hostPath": "/dev/dri/renderD134",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD135",
                        "hostPath": "/dev/dri/renderD135",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD136",
                        "hostPath": "/dev/dri/renderD136",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD137",
                        "hostPath": "/dev/dri/renderD137",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD138",
                        "hostPath": "/dev/dri/renderD138",
                        "permissions": [
                            "read",
                            "write"
                        ]
```

```
                    },
                    {
                        "containerPath": "/dev/dri/renderD139",
                        "hostPath": "/dev/dri/renderD139",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD140",
                        "hostPath": "/dev/dri/renderD140",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD141",
                        "hostPath": "/dev/dri/renderD141",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD142",
                        "hostPath": "/dev/dri/renderD142",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    },
                    {
                        "containerPath": "/dev/dri/renderD143",
                        "hostPath": "/dev/dri/renderD143",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ]
            },
            "mountPoints": [
```

```
                {
                    "containerPath": "/video",
                    "sourceVolume": "video_file"
                }
            ],
            "cpu": 0,
            "memory": 12000,
            "image": "0123456789012.dkr.ecr.us-west-2.amazonaws.com/aws/xilinx-
xffmpeg",
            "essential": true,
            "name": "xilinix-xffmpeg"
        }
    ],
    "volumes": [
        {
            "name": "video_file",
            "host": {"sourcePath": "/home/ec2-user"}
        }
    ]
}
```

2. Enregistrez la définition de tâche.

```
aws ecs register-task-definition --family vt1-24xlarge-xffmpeg-processor --cli-
input-json file://vt1-24xlarge-xffmpeg-linux.json --region us-east-1
```

Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones

Vous pouvez utiliser Amazon EC2 Trn 1, Amazon EC2 Trn2, Amazon EC2 Inf1 (Inf1 n'est pris en charge que sur le type de lancement EC2), et Amazon EC2 Inf2 instances avec vos clusters pour les charges de travail d'apprentissage automatique.

Les instances Amazon EC2 Trn1 et Trn2 sont alimentées par AWS Trainium chips. Ces instances fournissent une formation performante et peu coûteuse pour machine learning dans le cloud. Vous pouvez entraîner un modèle d'inférence d'apprentissage automatique à l'aide d'un framework d'apprentissage automatique avec AWS Neurone sur une instance Trn1 ou Trn2. Ensuite, vous pouvez exécuter le modèle sur une instance Inf1 (Inf1 n'est pris en charge que sur le type de lancement EC2), ou sur une instance Inf2 pour utiliser l'accélération du AWS Puces d'inférentialité.

Les instances Inf1 et Inf2 d'Amazon EC2 sont alimentées par des puces AWS Inferentia. Elles fournissent des performances élevées et les inférences de coûts les plus bas dans le cloud.

Les modèles de machine learning sont déployés sur des conteneurs à l'aide d'AWS Neuron, qui est un kit de développement logiciel (SDK) spécialisé. Le SDK comprend un compilateur, des outils d'exécution et de profilage qui optimisent les performances d'apprentissage automatique de AWS puces d'apprentissage automatique. AWS Neuron prend en charge les frameworks d'apprentissage automatique populaires tels que TensorFlow, PyTorchet Apache MXNet.

Considérations

Avant de commencer à déployer Neuron sur Amazon ECS, prenez en compte ce qui suit :

- Selon le type de lancement, vos clusters peuvent contenir un mélange de Trn1, Trn2, Inf1, Inf2 et d'autres instances.
- Vous avez besoin d'une application Linux dans un conteneur qui utilise un framework d'apprentissage automatique prenant en charge AWS Neurone.

> **Important**
>
> Les applications qui utilisent d'autres frameworks peuvent ne pas avoir de meilleures performances sur les instances Trn1, Trn2, Inf1 et Inf2.

- Amazon ECS prend en charge deux approches pour configurer l'accès aux appareils Neuron :
  - Allocation d'appareils Neuron gérée — Utilisez le `resourceRequirements` paramètre avec type `NeuronDevice` dans la définition de votre conteneur. Amazon ECS découvre et attribue automatiquement les appareils Neuron à vos conteneurs. Disponible uniquement sur les instances gérées. Pour plus d'informations, consultez Allocation d'appareils Neuron gérée.
  - Spécification manuelle du dispositif Neuron — Utilisez le `linuxParameters.devices` paramètre pour spécifier explicitement les chemins des périphériques Neuron. Disponible à la fois sur le type de lancement EC2 et sur les instances gérées. Pour plus d'informations, consultez Spécification manuelle du dispositif Neuron.

> **Important**
>
> Utilisez systématiquement une seule approche pour éviter les conflits.

Allocation d'appareils Neuron gérée

Avec les instances gérées, vous pouvez utiliser `resourceRequirements` paramètre dans votre définition de conteneur pour demander des appareils Neuron. Amazon ECS découvre automatiquement les appareils Neuron sur l'instance, les affecte à votre tâche et configure le conteneur avec accès à tous les appareils Neuron de l'instance. Comme la tâche nécessite un accès exclusif à tous les appareils, une seule tâche Neuron s'exécute par instance.

> **Note**
>
> `Inf1` les instances ne sont prises en charge que sur le type de lancement EC2. Pour utiliser les instances Inf1, consultez Spécification manuelle du dispositif Neuron.

Sélection d'instances de neurones

Pour sélectionner Neuron-enabled types d'instances pour vos charges de travail d'instances gérées, utilisez le `instanceRequirements` objet dans le modèle de lancement du fournisseur de capacité. Vous pouvez utiliser les attributs suivants pour sélectionner Neuron-enabled instances :

- `acceleratorManufacturers` — Utilisation `amazon-web-services` pour sélectionner des instances avec AWS accélérateurs (y compris Inferentia et Trainium).
- `acceleratorNames` — Utilisation `inferentia2`, `trainium`, ou `trainium2` pour sélectionner des puces accélératrices spécifiques.
- `allowedInstanceTypes` — Utilisation `inf*` et `trn*` pour sélectionner les types d'instances Neuron par leur nom.

L'exemple suivant utilise `allowedInstanceTypes` :

```
{
    "instanceRequirements": {
        "allowedInstanceTypes": ["inf*", "trn*"]
    }
}
```

Définition de tâche

Pour demander des appareils Neuron dans votre définition de tâche, ajoutez un `resourceRequirements` entrée avec type `NeuronDevice` et valeur `ALL`. Cela donne au conteneur un accès exclusif à tous les appareils Neuron de l'instance.

Les contraintes suivantes s’appliquent :

- Au plus une définition de conteneur peut spécifier `NeuronDevice` dans `resourceRequirements`.
- Vous ne pouvez pas combiner `resourceRequirements` avec type `NeuronDevice` et `linuxParameters.devices` pour les appareils Neuron dans la même définition de tâche.

Une fois votre tâche lancée, vous pouvez vérifier l'attribution de l'appareil Neuron en appelant le `DescribeTasks` Fonctionnement de l'API. La réponse comprend un `neuronDeviceIds` champ sur chaque conteneur qui affiche les identifiants des appareils Neuron attribués. Vous pouvez également appeler le `DescribeContainerInstances` Fonctionnement de l'API à afficher `NEURON_DEVICES` dans le `registeredResources` et `remainingResources` champs pour l'instance de conteneur.

Pour un exemple de définition de tâche, voir Exemples de définitions de tâches neuronales.

Spécification manuelle du dispositif Neuron

Avec cette approche, vous spécifiez manuellement AWS Trainium ou AWS Inferentia place les chemins du périphérique dans votre définition de tâche à l'aide du `linuxParameters.devices` paramètre. Cette approche fonctionne à la fois sur le type de lancement EC2 et sur les instances gérées.

Une seule tâche d'inférence ou d'entraînement d'inférence peut être exécutée sur chaque puce AWS Trainium ou AWS Inferentia. Vous pouvez exécuter autant de tâches qu'il y a de puces sur l'instance en affectant différents appareils à chaque tâche.

Pour le type de lancement EC2, vous pouvez utiliser les attributs du type d'instance lorsque vous configurez les contraintes de placement des tâches afin de vous assurer que la tâche est lancée sur le type d'instance que vous spécifiez. Pour plus d’informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

Exigences relatives à la définition de tâche

La définition de tâche doit être spécifique au type d'instance unique. Vous devez configurer un conteneur pour utiliser des AWS Trainium ou AWS Dispositifs d'inférence disponibles sur l'instance de conteneur hôte. Vous pouvez effectuer cette opération à l'aide du paramètre `linuxParameters`. Le tableau suivant détaille les cartes et les puces qui sont spécifiques à chaque type d'instance.

Type d'instance	vCPU	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
trn1.2xlarge	8	32	1	`/dev/neuron0`
trn1.32xlarge	128	512	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`

Type d'instance	vCPU	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
trn2.48xlarge	192	2048	16	`/dev/neuron0` , `/dev/neuron1` , `/dev/neuron2` , `/dev/neuron3` , `/dev/neuron4` , `/dev/neuron5` , `/dev/neuron6` , `/dev/neuron7` , `/dev/neuron8` , `/dev/neuron9` , `/dev/neuron10` , `/dev/neuron11` , `/dev/neuron12` , `/dev/neuron13` , `/dev/neuron14` , `/dev/neuron15`
inf1.xlarge	4	8	1	`/dev/neuron0`
inf1.2xlarge	8	16	1	`/dev/neuron0`

Type d'instance	vCPU	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
inf1.6xlarge	24	48	4	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3`
inf1.24xlarge	96	192	16	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11 ,/dev/neuron12 ,/dev/neuron13 ,/dev/neuron14 ,/dev/neuron15`

Type d'instance	vCPU	RAM (Gio)	AWS puces accélératrices ML	Chemins de l'appareil
inf2.xlarge	4	16	1	`/dev/neuron0`
inf2.8xlarge	32	128	1	`/dev/neuron0`
inf2.24xlarge	96	384	6	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5`
inf2.48xlarge	192	768	12	`/dev/neuron0 ,/dev/neuron1 ,/dev/neuron2 ,/dev/neuron3 ,/dev/neuron4 ,/dev/neuron5 ,/dev/neuron6 ,/dev/neuron7 ,/dev/neuron8 ,/dev/neuron9 ,/dev/neuron10 ,/dev/neuron11`

Pour un exemple de définition de tâche, voir Exemples de définitions de tâches neuronales.

Instances gérées

Les instances gérées utilisent automatiquement une AMI qui inclut le pilote Neuron. Aucune configuration d'AMI supplémentaire n'est requise.

Type de lancement EC2

Amazon ECS fournit une AMI optimisée pour Amazon ECS basée sur Amazon Linux 2023 pour AWS Trainium et AWS Charges de travail inférentielles. Il est livré avec AWS Pilotes Neuron et environnement d'exécution pour Docker. Cette AMI facilite l'exécution de charges de travail d'inférence de machine learning sur Amazon ECS.

Nous vous recommandons d'utiliser Amazon ECS-optimized AMI Amazon Linux 2023 (Neuron) lors du lancement de vos instances Amazon EC2 Trn1, Inf1 et Inf2.

Vous pouvez récupérer l'Amazon actuel ECS-optimized AMI Amazon Linux 2023 (Neuron) utilisant AWS CLI à l'aide de la commande suivante.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/recommended
```

Exemples de définitions de tâches neuronales

Exemple d'allocation d'appareils gérés

L'exemple suivant montre une définition de tâche qui demande à tous les appareils Neuron de l'instance à l'aide du `resourceRequirements` paramètre. Cette approche n'est disponible que sur les instances gérées.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["MANAGED_INSTANCES"],
    "networkMode": "awsvpc",
    "cpu": "8192",
    "memory": "16384",
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "name": "neuron-inference",
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/huggingface-vllm-inference-neuronx:0.11.0-optimum0.4.5-neuronx-py310-sdk2.26.1-ubuntu22.04",
```

```
            "essential": true,
            "command": [
                "--model", "${YOUR_HUGGING_FACE_MODEL_ID}",
                "--port", "8080",
                "--tensor-parallel-size", "2",
                "--allow-non-cached-model"
            ],
            "portMappings": [
                {
                    "containerPort": 8080,
                    "protocol": "tcp"
                }
            ],
            "resourceRequirements": [
                {
                    "type": "NeuronDevice",
                    "value": "ALL"
                }
            ]
        }
    ]
}
```

Dans cet exemple, l'image du conteneur inclut un serveur d'inférence vLLM optimisé pour AWS Neurone. Le point d'entrée de l'image télécharge un modèle depuis HuggingFace, le compile pour Neuron et lance un OpenAI-compatible Serveur API sur le port 8080. Remplacer *${YOUR_HUGGING_FACE_MODEL_ID}* avec votre HuggingFace ID du modèle.

Exemple de spécification manuelle de l'appareil

L'exemple suivant montre une définition de tâche Linux pour `inf1.xlarge` en utilisant le type de lancement EC2 avec `linuxParameters.devices` pour spécifier les chemins des périphériques Neuron.

```
{
    "family": "ecs-neuron",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
```

```
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == inf1.xlarge"
        }
    ],
    "executionRoleArn": "${YOUR_EXECUTION_ROLE}",
    "containerDefinitions": [
        {
            "entryPoint": [
                "/usr/local/bin/entrypoint.sh",
                "--port=8500",
                "--rest_api_port=9000",
                "--model_name=resnet50_neuron",
                "--model_base_path=s3://amzn-s3-demo-bucket/resnet50_neuron/"
            ],
            "portMappings": [
                {
                    "hostPort": 8500,
                    "protocol": "tcp",
                    "containerPort": 8500
                },
                {
                    "hostPort": 8501,
                    "protocol": "tcp",
                    "containerPort": 8501
                },
                {
                    "hostPort": 0,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "linuxParameters": {
                "devices": [
                    {
                        "containerPath": "/dev/neuron0",
                        "hostPath": "/dev/neuron0",
                        "permissions": [
                            "read",
                            "write"
                        ]
                    }
                ],
                "capabilities": {
                    "add": [
```

```
                        "IPC_LOCK"
                    ]
                }
            },
            "cpu": 0,
            "memoryReservation": 1000,
            "image": "763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-
inference-neuron:1.15.4-neuron-py37-ubuntu18.04",
            "essential": true,
            "name": "resnet50"
        }
    ]
}
```

Définitions de tâche Amazon ECS pour les instances de deep learning

Pour utiliser des charges de travail Deep Learning sur Amazon ECS, enregistrez des instances Amazon EC2 DL1 dans vos clusters. Les instances Amazon EC2 DL1 sont alimentées par des accélérateurs Gaudi de Habana Labs (une société Intel). Utilisez le SDK Habana SynapseAI pour vous connecter aux accélérateurs Habana Gaudi. Le SDK prend en charge les frameworks d'apprentissage automatique les plus populaires, TensorFlow et PyTorch.

Considérations

Avant de commencer à déployer DL1 sur Amazon ECS, prenez en compte ce qui suit :

- Vos clusters peuvent contenir une combinaison d'instances DL1 et non DL1.
- Lorsque vous exécutez une tâche autonome ou créez un service, vous pouvez utiliser des attributs de type d'instance lors de la configuration des contraintes de placement des tâches afin de garantir sur quelles instances de conteneur la tâche, que vous spécifiez, est lancée. Cela garantit que vos ressources sont utilisées efficacement et que vos tâches pour les charges de travail de deep learning se trouvent sur vos instances DL1. Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

  L'exemple suivant exécute une tâche sur une instance `dl1.24xlarge` de votre cluster `default`.

```
aws ecs run-task \
     --cluster default \
     --task-definition ecs-dl1-task-def \
```

```
    --placement-constraints type=memberOf,expression="attribute:ecs.instance-type ==
dl1.24xlarge"
```

Utilisation d'une AMI DL1

Vous avez 3 options pour exécuter une AMI sur des instances Amazon EC2 DL1 pour Amazon ECS :

- AWS Marketplace AMI fournies par Habana ici.
- Il s'agit des AMI de deep learning Habana fournies par Amazon Web Services. Comme il n'est pas inclus, vous devez installer l'agent de conteneur Amazon ECS séparément.
- Utilisez Packer pour créer une AMI personnalisée fournie par GitHub repo. Pour plus d'informations, consultez la section documentation Packer.

Spécification du deep learning dans une définition de tâche Amazon ECS

Pour exécuter des conteneurs d'apprentissage profond accélérés Habana Gaudi sur Amazon ECS, votre définition de tâche doit contenir la définition du conteneur d'un conteneur préconstruit qui utilise le modèle d'apprentissage en profondeur pour TensorFlow ou PyTorch en utilisant Habana SynapseAI fourni par AWS Conteneurs de Deep Learning.

L'image de conteneur suivante contient TensorFlow 2.7.0 et Ubuntu 20.04. Une liste complète des conteneurs Deep Learning prédéfinis et optimisés pour les accélérateurs Habana Gaudi est mise à jour sur GitHub. Pour de plus amples informations, veuillez consulter la section Conteneurs d'entraînement Habana.

```
763104351884.dkr.ecr.us-east-1.amazonaws.com/tensorflow-training-habana:2.7.0-hpu-py38-
synapseai1.2.0-ubuntu20.04
```

Voici un exemple de définition de tâche de conteneurs Linux sur Amazon EC2 qui affiche la syntaxe à utiliser. Cet exemple utilise une image contenant l'outil d'interface de gestion du système Habana Labs (HL-SMI) trouvé ici : `vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/tensorflow-installer-tf-cpu-2.6.0:1.1.0-614`

```
{
    "family": "dl-test",
    "requiresCompatibilities": ["EC2"],
    "placementConstraints": [
```

```
        {
            "type": "memberOf",
            "expression": "attribute:ecs.os-type == linux"
        },
        {
            "type": "memberOf",
            "expression": "attribute:ecs.instance-type == dl1.24xlarge"
        }
    ],
    "networkMode": "host",
    "cpu": "10240",
    "memory": "1024",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "command": ["hl-smi"],
            "cpu": 8192,
            "environment": [
                {
                    "name": "HABANA_VISIBLE_DEVICES",
                    "value": "all"
                }
            ],
            "image": "vault.habana.ai/gaudi-docker/1.1.0/ubuntu20.04/habanalabs/
tensorflow-installer-tf-cpu-2.6.0:1.1.0-614",
            "essential": true,
            "name": "tensorflow-installer-tf-hpu"
        }
    ]
}
```

Définitions de tâche Amazon ECS pour les charges de travail ARM 64 bits

Amazon ECS prend en charge l'utilisation d'applications ARM 64 bits. Vous pouvez exécuter vos applications sur la plateforme optimisée par les processeurs AWS Graviton. Cette plateforme est adaptée à une grande variété de charges de travail. Cela inclut les charges de travail telles que les serveurs d'applications, les microservices, le calcul à haute performance, CPU-based inférence d'apprentissage automatique, encodage vidéo, automatisation de la conception électronique, jeux, bases de données open source et caches en mémoire.

Considérations

Avant de commencer à déployer des définitions de tâche utilisant l'architecture ARM 64 bits, tenez compte des points suivants :

- Les applications peuvent utiliser Fargate ou EC2.
- Les applications ne peuvent utiliser que le système d'exploitation Linux.
- Pour le type Fargate, les applications doivent utiliser la version de plateforme `1.4.0` Fargate ou version ultérieure.
- Les applications peuvent utiliser Fluent Bit ou CloudWatch pour la surveillance.
- Pour le Fargate, ce qui suit Régions AWS ne prennent pas en charge les charges de travail ARM 64 bits :
  - USA Est (Virginie du Nord), la zone de disponibilité `use1-az3`
- Pour EC2, consultez les points suivants pour vérifier que votre Région prend en charge le type d'instance que vous souhaitez utiliser :
  - Instances M6g Amazon EC2
  - Instances T4g Amazon EC2
  - Instances C6g Amazon EC2
  - Instances R6gd Amazon EC2
  - Instances X2gd Amazon EC2

  Vous pouvez également utiliser la commande `describe-instance-type-offerings` Amazon EC2 avec un filtre pour afficher l'offre d'instances pour votre Région.

  ```
  aws ec2 describe-instance-type-offerings --filters Name=instance-
  type,Values=instance-type --region region
  ```

  L'exemple suivant vérifie la disponibilité du type d'instance M6 dans la Région USA Est (Virginie du Nord) (us-east-1).

  ```
  aws ec2 describe-instance-type-offerings --filters "Name=instance-type,Values=m6*" --
  region us-east-1
  ```

  Pour plus d'informations, consultez describe-instance-type-offerings dans la Référence des commandes en ligne Amazon EC2.

Spécification de l’architecture ARM dans votre définition de tâche Amazon ECS

Pour utiliser l'architecture ARM, spécifiez `ARM64` pour paramètre de la définition de la tâche `cpuArchitecture`.

Dans l'exemple suivant, l'architecture ARM est spécifiée dans une définition de tâche. Elles sont au format JSON.

```
{
    "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
    },
...
}
```

Dans l'exemple suivant, une définition de tâche pour l'architecture ARM affiche « hello world ».

```
{
 "family": "arm64-testapp",
 "networkMode": "awsvpc",
 "containerDefinitions": [
    {
        "name": "arm-container",
        "image": "public.ecr.aws/docker/library/busybox:latest",
        "cpu": 100,
        "memory": 100,
        "essential": true,
        "command": [ "echo hello world" ],
        "entryPoint": [ "sh", "-c" ]
    }
 ],
 "requiresCompatibilities": [ "EC2" ],
 "cpu": "256",
 "memory": "512",
 "runtimePlatform": {
        "operatingSystemFamily": "LINUX",
        "cpuArchitecture": "ARM64"
  },
 "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole"
}
```

Envoyer les journaux Amazon ECS à CloudWatch

Vous pouvez configurer les conteneurs dans vos tâches pour envoyer les informations de journal à CloudWatch Journaux. Si vous utilisez Fargate pour vos tâches, vous pouvez afficher les journaux de vos conteneurs. Si vous utilisez EC2, vous pouvez afficher différents journaux de vos conteneurs dans un emplacement pratique, et cela empêche vos journaux de conteneur d'occuper de l'espace disque sur vos instances de conteneur.

> **Note**
>
> Le type des informations qui sont consignées par les conteneurs dans votre tâche dépend principalement de leur commande `ENTRYPOINT`. Par défaut, les journaux capturés affichent la sortie de commande que vous pouvez généralement voir dans un terminal interactif si vous exécutiez le conteneur localement, qui sont les `STDOUT` et `STDERR` I/O cours d'eau. Le `awslogs` log driver transmet simplement ces journaux de Docker à CloudWatch Journaux. Pour plus d'informations sur la façon dont Docker traite les journaux, y compris les autres moyens de capturer différents fichiers, données ou flux, consultez Afficher les journaux d'un conteneur ou d'un service dans la documentation Docker.

Pour envoyer les journaux système de vos instances de conteneur Amazon ECS à CloudWatch Journaux, voir Fichiers journaux de surveillance et CloudWatch Quotas de journaux dans le Amazon CloudWatch Guide de l'utilisateur des journaux.

Fargate

Si vous utilisez Fargate pour vos tâches, vous devez ajouter les paramètres `logConfiguration` requis à votre définition de tâche pour activer le pilote de journalisation `awslogs`. Pour plus d'informations, consultez Exemple de définition de tâche Amazon ECS : acheminer les journaux vers CloudWatch.

Pour le conteneur Windows sur Fargate, effectuez l'une des options suivantes lorsque l'un des paramètres de définition de tâche comporte des caractères spéciaux tels que `& \ < > ^ |` :

- Ajoutez un caractère d'échappement (\) avec des guillemets autour de la chaîne de paramètres entière

  Exemple

```
"awslogs-multiline-pattern": "\"^[|DEBUG|INFO|WARNING|ERROR\"",
```

- Ajoutez un caractère d'échappement (^) autour de chaque caractère spécial

  Exemple

```
"awslogs-multiline-pattern": "^^[^|DEBUG^|INFO^|WARNING^|ERROR",
```

EC2

Si vous utilisez le type de lancement EC2 pour vos tâches et que vous souhaitez activer le pilote de journalisation `awslogs`, vos instances de conteneur Amazon ECS nécessitent au moins la version 1.9.0 de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous devez utiliser soit un Amazon ECS-optimized AMI ou AMI personnalisée avec au moins une version `1.9.0-1` de la `ecs-init` colis. Lorsque vous utilisez une AMI personnalisée, vous devez spécifier que le pilote de journalisation `awslogs` est disponible sur l'instance Amazon EC2 lorsque vous démarrez l'agent en utilisant la variable d'environnement suivante dans votre instruction docker run ou votre fichier de variables d'environnement.
>
> ```
> ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
> ```

Vos instances de conteneur Amazon ECS nécessitent également une autorisation `logs:CreateLogStream` et `logs:PutLogEvents` sur le rôle IAM avec lequel vous pouvez lancer vos instances de conteneur. Si vous avez créé votre rôle d'instance de conteneur Amazon ECS avant l'activation de la prise en charge du pilote du journal `awslogs` dans Amazon ECS, vous devrez sans doute ajouter ces autorisations. `ecsTaskExecutionRole` est utilisé lorsqu'il est affecté à la tâche et contient probablement les autorisations appropriées. Pour plus d'informations sur le rôle d'exécution de tâche, consultez la section Rôle IAM d'exécution de tâche Amazon ECS. Si vos instances de conteneur utilisent la politique IAM gérée pour les instances de conteneur, vos instances de conteneur disposent probablement des autorisations appropriées. Pour obtenir des informations

sur la politique IAM gérée pour les instances de conteneur, consultez la section Rôle IAM d'instance de conteneur Amazon ECS.

Exemple de définition de tâche Amazon ECS : acheminer les journaux vers CloudWatch

Avant que vos conteneurs puissent envoyer des journaux à CloudWatch, vous devez spécifier `awslogs` enregistrez le pilote pour les conteneurs dans votre définition de tâche. Pour plus d'informations sur les paramètres de journalisation, consultez la section Stockage et journalisation.

La définition de tâche JSON qui suit possède un objet `logConfiguration` spécifié pour chaque conteneur. L'un est pour WordPress conteneur qui envoie des journaux à un groupe de journaux appelé `awslogs-wordpress`. L'autre concerne un conteneur MySQL qui envoie des journaux à un groupe de journaux appelé `awslogs-mysql`. Les deux conteneurs utilisent le préfixe de flux de journal `awslogs-example`.

```
{
    "containerDefinitions": [
        {
            "name": "wordpress",
            "links": [
                "mysql"
            ],
            "image": "public.ecr.aws/docker/library/wordpress:latest",
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-wordpress",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example"
                }
            },
            "memory": 500,
            "cpu": 10
```

```
        },
        {
            "environment": [
                {
                    "name": "MYSQL_ROOT_PASSWORD",
                    "value": "password"
                }
            ],
            "name": "mysql",
            "image": "public.ecr.aws/docker/library/mysql:latest",
            "cpu": 10,
            "memory": 500,
            "essential": true,
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-create-group": "true",
                    "awslogs-group": "awslogs-mysql",
                    "awslogs-region": "us-west-2",
                    "awslogs-stream-prefix": "awslogs-example",
                    "mode": "non-blocking",
                    "max-buffer-size": "25m"
                }
            }
        }
    ],
    "family": "awslogs-example"
}
```

Étapes suivantes

- Vous pouvez éventuellement définir une politique de conservation pour le groupe de journaux à l'aide du CloudWatch AWS CLI ou API. Pour plus d'informations, consultez la section put-retention-policy dans la Référence des commandes de l'AWS Command Line Interface .
- Après avoir enregistré une définition de tâche auprès du `awslogs` pilote de journal dans une configuration de journal de définition de conteneur, vous pouvez exécuter une tâche ou créer un service avec cette définition de tâche pour commencer à envoyer des journaux à CloudWatch Journaux. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS et Création d'un déploiement de mise à jour propagée Amazon ECS.

Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner

Vous pouvez utiliser FireLens pour qu'Amazon ECS utilise les paramètres de définition des tâches pour acheminer les journaux vers un AWS service ou AWS Partner Network Destination (APN) pour le stockage et l'analyse des journaux. Le AWS Partner Network est une communauté mondiale de partenaires qui s'appuie sur des programmes, une expertise et des ressources pour créer, commercialiser et vendre des offres aux clients. Pour plus d'informations, veuillez consulter AWS Partner. FireLens fonctionne avec Fluentd et Fluent Bit. Nous fournissons l'image AWS pour Fluent Bit ou vous pouvez utiliser votre propre image Fluentd ou Fluent Bit.

Par défaut, Amazon ECS configure la dépendance du conteneur pour que le conteneur Firelens démarre avant tout conteneur qui l'utilise. Le conteneur Firelens s'arrête également une fois que tous les conteneurs qui l'utilisent se sont arrêtés.

Pour utiliser cette fonctionnalité, vous devez créer un rôle IAM pour vos tâches qui fournit les autorisations nécessaires pour utiliser tout AWS les services requis par les tâches. Par exemple, si un conteneur achemine des journaux vers Firehose, la tâche nécessite l'autorisation d'appeler l'API `firehose:PutRecordBatch`. Pour plus d'informations, consultez Ajout et suppression d'autorisations basées sur l'identité IAM dans le Guide de l'utilisateur IAM.

Votre tâche peut également nécessiter le rôle d'exécution de tâche Amazon ECS dans les conditions suivantes. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

- Si votre tâche est hébergée sur Fargate et que vous extrayez des images de conteneurs depuis Amazon ECR ou que vous faites référence à des données sensibles depuis AWS Secrets Manager dans la configuration de votre journal, vous devez inclure le rôle IAM d'exécution des tâches.
- Lorsque vous utilisez un fichier de configuration personnalisé hébergé dans Amazon S3, votre rôle IAM d'exécution de tâche doit inclure l'autorisation `s3:GetObject`.

Tenez compte des points suivants lors de l'utilisation FireLens pour Amazon ECS :

- Nous vous recommandons d'ajouter `my_service_` au nom du conteneur de journaux afin de pouvoir facilement le distinguer dans la console.
- Amazon ECS ajoute par défaut une dépendance relative à l'ordre des conteneurs de démarrage entre les conteneurs d'applications et le conteneur FireLens. Lorsque vous spécifiez un ordre de conteneur entre les conteneurs d'applications et le conteneur FireLens, l'ordre de conteneur de départ par défaut est remplacé.

- FireLens pour Amazon ECS est pris en charge pour les tâches hébergées à la fois sur AWS Fargate sur Linux et Amazon EC2 sur Linux. Les conteneurs Windows ne prennent pas en charge FireLens.

  Pour plus d'informations sur la configuration de la journalisation centralisée pour les conteneurs Windows, consultez la section Journalisation centralisée pour les conteneurs Windows sur Amazon ECS à l'aide de Fluent Bit.

- Vous pouvez utiliser CloudFormation modèles à configurer FireLens pour Amazon ECS. Pour plus d'informations, voir AWS : :ECS :TaskDefinition FirelensConfiguration dans le AWS CloudFormation Guide de l'utilisateur

- FireLens écoute sur le port `24224`, afin de garantir que le FireLens log router n'est pas accessible en dehors de la tâche, vous ne devez pas autoriser le trafic entrant sur le port `24224` dans le groupe de sécurité utilisé par votre tâche. Pour les tâches utilisant le mode réseau `awsvpc`, il s'agit du groupe de sécurité associé à la tâche. Pour les tâches utilisant le mode réseau `host`, il s'agit du groupe de sécurité associé à l'instance Amazon EC2 qui héberge la tâche. Pour les tâches utilisant le mode réseau `bridge`, ne créez pas de mappages de ports utilisant le port `24224`.

- Pour les tâches qui utilisent le `bridge` mode réseau, le conteneur avec FireLens la configuration doit démarrer avant que les conteneurs d'applications qui en dépendent ne démarrent. Pour contrôler l'ordre de début de vos conteneurs, utilisez les conditions de dépendance dans la définition de tâche. Pour plus d'informations, consultez Dépendances du conteneur.

  > **Note**
  >
  > Si vous utilisez des paramètres de condition de dépendance dans les définitions de conteneurs avec FireLens configuration, assurez-vous que chaque conteneur possède un `START` ou `HEALTHY` condition requise.

- Par défaut, FireLens ajoute le nom de la définition du cluster et de la tâche ainsi que l'Amazon Resource Name (ARN) du cluster en tant que clés de métadonnées à votre stdout/stderr journaux de conteneurs. Voici un exemple du format de métadonnées.

```
"ecs_cluster": "cluster-name",
"ecs_task_arn": "arn:aws:ecs:region:111122223333:task/cluster-
name/f2ad7dba413f45ddb4EXAMPLE",
"ecs_task_definition": "task-def-name:revision",
```

Si vous ne souhaitez pas que les métadonnées figurent dans vos journaux, définissez `enable-ecs-log-metadata` sur `false`dans la section `firelensConfiguration` de la définition de tâche.

```
"firelensConfiguration":{
   "type":"fluentbit",
   "options":{
      "enable-ecs-log-metadata":"false",
      "config-file-type":"file",
      "config-file-value":"/extra.conf"
}
```

Vous pouvez configurer le FireLens conteneur à exécuter en tant qu'utilisateur non root. Éléments à prendre en compte :

- Pour configurer le FireLens conteneur à exécuter en tant qu'utilisateur non root, vous devez spécifier l'utilisateur dans l'un des formats suivants :
  - `uid`
  - `uid:gid`
  - `uid:group`

  Pour plus d'informations sur la spécification d'un utilisateur dans une définition de conteneur, consultez ContainerDefinition dans le Référence de l'API Amazon Elastic Container Service.

  Le FireLens le conteneur reçoit les journaux des applications via un UNIX douille. L'agent Amazon ECS utilise `uid` pour attribuer la propriété du répertoire des sockets au FireLens contenant.
- Configuration du FireLens le conteneur à exécuter en tant qu'utilisateur non root est pris en charge sur la version Amazon ECS Agent `1.96.0` et versions ultérieures, et Amazon ECS-optimized Version AMI `v20250716` et plus tard.
- Lorsque vous spécifiez un utilisateur pour FireLens récipient, le `uid` doit être unique et ne pas être utilisé pour d'autres processus appartenant à d'autres conteneurs de la tâche ou de l'instance de conteneur.

Pour plus d’informations sur l’utilisation de plusieurs fichiers de configuration avec Amazon ECS, y compris les fichiers que vous hébergez ou les fichiers dans Amazon S3, consultez la section Processus d’initialisation pour Fluent Bit sur ECS, support multiconfigurations.

Pour plus d'informations sur les exemples de configurations, voir Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

Pour plus d'informations sur la configuration des journaux pour un débit élevé, voir Configuration des journaux Amazon ECS pour un débit élevé.

Configuration des journaux Amazon ECS pour un débit élevé

Pour les scénarios de débit de journalisation élevé, nous vous recommandons d'utiliser le `awsfirelens` pilote de journal avec FireLens et Fluent Bit. Fluent Bit est un processeur de journaux léger qui utilise efficacement les ressources et peut gérer des millions d'enregistrements de journaux. Cependant, pour obtenir des performances optimales à grande échelle, il faut ajuster sa configuration.

Cette section couvre les sujets avancés Fluent Bit techniques d'optimisation permettant de gérer un débit de journalisation élevé tout en préservant la stabilité du système et en évitant toute perte de données.

Pour plus d'informations sur l'utilisation des fichiers de configuration personnalisés avec FireLens, voir Utilisation d'un fichier de configuration. Pour d'autres exemples, voir Amazon ECS FireLens exemples sur GitHub.

> **Note**
>
> Certaines options de configuration de cette section, telles que `workers` et `threaded`, exiger AWS pour Fluent Bit version 3 ou ultérieure. Pour plus d'informations sur les versions disponibles, voir AWS pour les versions de Fluent Bit.

Comprendre les morceaux

Fluent Bit traite les données dans des unités appelées morceaux. Lorsqu'un plug-in INPUT reçoit des données, le moteur crée un bloc qui est stocké en mémoire ou sur le système de fichiers avant d'être envoyé aux destinations OUTPUT.

Le comportement de mise en mémoire tampon dépend de `storage.type` réglage dans vos sections INPUT. Par défaut, Fluent Bit utilise la mémoire tampon. Pour les scénarios de haut débit ou de production, la mise en mémoire tampon du système de fichiers offre une meilleure résilience.

Pour plus d'informations, voir Morceaux dans le Fluent Bit documentation et Qu'est-ce qu'un Chunk ? dans le AWS pour Fluent Bit référentiel d'exemples.

Mise en mémoire tampon (par défaut)

Par défaut, Fluent Bit utilise la mémoire tampon (`storage.type memory`). Vous pouvez limiter l'utilisation de la mémoire par plug-in INPUT à l'aide du `Mem_Buf_Limit` paramètre.

L'exemple suivant montre une configuration d'entrée avec mémoire tampon :

```
[INPUT]
    Name          tcp
    Tag           ApplicationLogs
    Port          5170
    storage.type  memory
    Mem_Buf_Limit 5MB
```

Important

Quand `Mem_Buf_Limit` est dépassé pour un plugin, Fluent Bit interrompt la saisie et les nouveaux enregistrements sont perdus. Cela peut provoquer une contre-pression et ralentir votre application. L'avertissement suivant apparaît dans Fluent Bit journaux :

```
[input] tcp.1 paused (mem buf overlimit)
```

La mise en mémoire tampon convient aux cas d'utilisation simples avec un débit de journalisation faible à modéré. Pour les scénarios de haut débit ou de production où la perte de données est un problème, utilisez plutôt la mise en mémoire tampon du système de fichiers.

Pour plus d'informations, voir Mise en mémoire tampon et mémoire dans le Fluent Bit documentation et Mise en mémoire tampon uniquement dans le AWS pour Fluent Bit référentiel d'exemples.

Mise en mémoire tampon du système de fichiers

Pour les scénarios à haut débit, nous vous recommandons d'utiliser la mise en mémoire tampon du système de fichiers. Pour plus d'informations sur la façon dont Fluent Bit gère la mise en mémoire tampon et le stockage, voir Mise en mémoire tampon et stockage dans le Fluent Bit documentation.

La mise en mémoire tampon du système de fichiers présente les avantages suivants :

- Plus grande capacité de mémoire tampon — L'espace disque est généralement plus abondant que la mémoire.

- Persistance — Les données mises en mémoire tampon survivent Fluent Bit redémarre.
- Dégradation gracieuse — En cas de panne de sortie, les données s'accumulent sur le disque au lieu d'épuiser la mémoire.

Pour activer la mise en mémoire tampon du système de fichiers, fournissez un Fluent Bit fichier de configuration. L'exemple suivant montre la configuration recommandée :

```
[SERVICE]
    # Flush logs every 1 second
    Flush 1
    # Wait 120 seconds during shutdown to flush remaining logs
    Grace 120
    # Directory for filesystem buffering
    storage.path             /var/log/flb-storage/
    # Limit chunks stored 'up' in memory (reduce for memory-constrained environments)
    storage.max_chunks_up    32
    # Flush backlog chunks to destinations during shutdown (prevents log loss)
    storage.backlog.flush_on_shutdown On

[INPUT]
    Name forward
    unix_path /var/run/fluent.sock
    # Run input in separate thread to prevent blocking
    threaded true
    # Enable filesystem buffering for persistence
    storage.type filesystem

[OUTPUT]
    Name cloudwatch_logs
    Match *
    region us-west-2
    log_group_name /aws/ecs/my-app
    log_stream_name $(ecs_task_id)
    # Use multiple workers for parallel processing
    workers 2
    # Retry failed flushes up to 15 times
    retry_limit 15
    # Maximum disk space for buffered data for this output
    storage.total_limit_size 10G
```

Paramètres de configuration clés :

`storage.path`

Le répertoire où Fluent Bit stocke les morceaux mis en mémoire tampon sur le disque.

`storage.backlog.flush_on_shutdown`

Lorsque cette option est activée, Fluent Bit tente de vider tous les segments du système de fichiers en attente vers leur destination lors de l'arrêt. Cela permet de garantir la livraison des données avant Fluent Bit s'arrête, mais peut augmenter le temps d'arrêt.

`storage.max_chunks_up`

Le nombre de morceaux qui restent en mémoire. La valeur par défaut est de 128 blocs, qui peuvent consommer plus de 500 Mo de mémoire car chaque bloc peut utiliser jusqu'à 4 à 5 Mo. Dans les environnements où la mémoire est limitée, réduisez cette valeur. Par exemple, si 50 Mo sont disponibles pour la mise en mémoire tampon, réglez-le sur 8 à 10 segments.

`storage.type filesystem`

Active le stockage du système de fichiers pour le plug-in d'entrée. Malgré son nom, Fluent Bit les usages `mmap` pour mapper des segments à la fois à la mémoire et au disque, garantissant ainsi la persistance sans sacrifier les performances.

`storage.total_limit_size`

L'espace disque maximal pour les données mises en mémoire tampon pour un plug-in OUTPUT spécifique. Lorsque cette limite est atteinte, les enregistrements les plus anciens pour cette sortie sont supprimés. Pour plus d'informations sur les tailles, voir Compréhension `stockage.total_limit_size`.

`threaded true`

Exécute l'entrée dans son propre thread, séparé de Fluent Bitde la boucle des événements principaux. Cela empêche les entrées lentes de bloquer l'ensemble du pipeline.

Pour plus d'informations, voir Mise en mémoire tampon du système de fichiers dans le Fluent Bit documentation et Système de fichiers et mise en mémoire tampon dans le AWS pour Fluent Bit référentiel d'exemples.

Compréhension **`stockage.total_limit_size`**

Le `storage.total_limit_size` le paramètre de chaque plug-in OUTPUT contrôle l'espace disque maximal pour les données mises en mémoire tampon pour cette sortie. Lorsque cette limite

est atteinte, les enregistrements les plus anciens pour cette sortie sont supprimés pour faire de la place pour de nouvelles données. Lorsque l'espace disque est complètement épuisé, Fluent Bit ne met pas les enregistrements en file d'attente et ils sont perdus.

Utilisez la formule suivante pour calculer la valeur appropriée `storage.total_limit_size` en fonction de votre taux de journalisation et de la fenêtre de restauration souhaitée :

```
If log rate is in KB/s, convert to MB/s first:
log_rate (MB/s) = log_rate (KB/s) / 1000

storage.total_limit_size (GB) = log_rate (MB/s) × duration (hours) × 3600 (seconds/
hour) / 1000 (MB to GB)
```

Le tableau suivant présente des exemples de calculs pour les taux de journalisation et les fenêtres de restauration courants :

Taux de journalisation	1 heure	6 heures	12 heures	24 heures
0,25 MB/s	0,9 GO	5,4 GO	10,8 GO	21,6 GO
0,5 MB/s	1,8 GO	10,8 GO	21,6 GO	43,2 GO
1 MB/s	3,6 GO	21,6 GO	43,2 GO	86,4 GO
5 MB/s	18 GO	108 GO	216 GO	432 GO
10 MB/s	36 GO	216 GO	432 GO	864 GO

Pour observer le débit maximal et choisir les tailles de mémoire tampon appropriées, utilisez le débit de mesure FireLens échantillon.

Utilisez la formule, des exemples de calculs et des analyses comparatives pour choisir une solution appropriée `storage.total_limit_size` qui fournit une piste pour une reprise optimale en cas de panne.

Exigences en matière de stockage des tâches Amazon ECS

Tout additionner `storage.total_limit_size` valeurs dans les sections OUTPUT et ajoutez une mémoire tampon pour les frais généraux. Ce total détermine l'espace de stockage nécessaire dans

votre définition de tâche Amazon ECS. Par exemple, 3 sorties × 10 Go chacune = 30 Go + mémoire tampon (5 à 10 Go) = 35 à 40 Go au total requis. Si le total dépasse l'espace de stockage disponible, Fluent Bit peut ne pas mettre les enregistrements en file d'attente et ils seront perdus.

Les options de stockage suivantes sont disponibles :

Supports de liaison (stockage éphémère)

- Pour AWS Fargate, la valeur par défaut est de 20 Go de stockage éphémère (maximum 200 Go). Configurer à l'aide de `ephemeralStorage` dans la définition de la tâche. Pour plus d'informations, consultez EphemeralStorage dans le Guide de l'utilisateur AWS CloudFormation .
- Pour EC2, la valeur par défaut est de 30 Go lorsque vous utilisez Amazon ECS-optimized AMI (partagée entre le système d'exploitation et Docker). Augmentez en modifiant la taille du volume de la racine.

Volumes Amazon EBS

- Fournit un stockage par blocs hautement disponible, durable et performant.
- Nécessite une configuration de volume et `mountPoint` dans la définition de la tâche pointant vers `storage.path` (par défaut : `/var/log/flb-storage/`).
- Pour plus d'informations, consultez Report de la configuration du volume au moment du lancement dans une définition de tâche Amazon ECS.

Volumes Amazon EFS

- Fournit un stockage de fichiers simple et évolutif.
- Nécessite une configuration de volume et `mountPoint` dans la définition de la tâche pointant vers `storage.path` (par défaut : `/var/log/flb-storage/`).
- Pour plus d'informations, consultez Spécification d'un système de fichiers Amazon EFS dans une définition de tâche Amazon ECS.

Pour plus d'informations sur les volumes de données, voir Options de stockage pour les tâches Amazon ECS.

Optimisation de la configuration des sorties

Les problèmes de réseau, les pannes de service et la limitation des destinations peuvent empêcher la distribution des journaux. Une configuration de sortie appropriée garantit la résilience sans perte de données.

En cas d'échec d'un rinçage de sortie, Fluent Bit peut recommencer l'opération. Les paramètres suivants contrôlent le comportement des nouvelles tentatives :

`retry_limit`

Nombre maximum de tentatives après la première tentative avant de supprimer des enregistrements. La valeur par défaut est 1. Par exemple, `retry_limit 3` signifie 4 tentatives au total (1 tentative initiale + 3 nouvelles tentatives). Pour les environnements de production, nous recommandons 15 ou plus, ce qui permet de couvrir plusieurs minutes de panne avec un retard exponentiel.

Réglez sur `no_limits` ou `False` pour une infinité de tentatives :

- Avec la mise en mémoire tampon, des tentatives infinies provoquent une pause du plug-in d'entrée lorsque les limites de mémoire sont atteintes.
- Avec la mise en mémoire tampon du système de fichiers, les enregistrements les plus anciens sont supprimés lorsque `storage.total_limit_size` est atteint.

> **Important**
>
> Après avoir épuisé toutes les tentatives de nouvelle tentative (1 initiale + `retry_limit` nouvelles tentatives), les enregistrements sont supprimés. AWS plugins avec `auto_retry_requests true` (par défaut) fournir une couche de nouvelle tentative supplémentaire avant Fluent Bitmécanisme de nouvelle tentative. Pour plus d'informations, voir Configurer les nouvelles tentatives dans le Fluent Bit documentation. Par exemple, `retry_limit 3` avec les paramètres par défaut (`scheduler.base 5`, `scheduler.cap 2000`, `net.connect_timeout 10s`) fournit environ 70 secondes de temps d'attente au planificateur (10 s + 20 s + 40 s), 40 secondes de délais de connexion réseau (4 tentatives × 10 s), plus AWS nouvelles tentatives de plug-in : durée totale d'environ 2 à 10 minutes selon les conditions du réseau et les délais TCP du système d'exploitation.

`scheduler.base`

Les secondes de base entre les nouvelles tentatives (par défaut : 5). Nous recommandons 10 secondes.

`scheduler.cap`

Le nombre maximum de secondes entre les nouvelles tentatives (par défaut : 2000). Nous recommandons 60 secondes.

Le temps d'attente entre les nouvelles tentatives utilise un retard exponentiel avec gigue :

```
wait_time = random(base, min(base × 2^retry_number, cap))
```

Par exemple, avec `scheduler.base 10` et `scheduler.cap 60`:

- Première tentative : attente aléatoire entre 10 et 20 secondes
- Deuxième tentative : attente aléatoire entre 10 et 40 secondes
- Troisième tentative et plus tard : attente aléatoire entre 10 et 60 secondes (plafonnée)

Pour plus d'informations, voir Configurer le temps d'attente pour une nouvelle tentative et Réseautage dans le Fluent Bit documentation.

`workers`

Le nombre de threads pour le traitement des sorties parallèles. Plusieurs serveurs autorisent des vidanges simultanées, améliorant ainsi le débit lors du traitement de nombreux blocs.

`auto_retry_requests`

Un AWS paramètre spécifique au plugin qui fournit une couche de nouvelle tentative supplémentaire avant Fluent Bitmécanisme de nouvelle tentative intégré. La valeur par défaut est `true`. Lorsque cette option est activée, AWS le plugin de sortie réessaie les demandes ayant échoué en interne avant que la demande ne soit considérée comme un échec de vidage et sous réserve de `retry_limit` configuration.

Le `Grace` paramètre dans le `[SERVICE]` la section définit l'heure Fluent Bit attend pendant l'arrêt pour vider les données mises en mémoire tampon. Le `Grace` la période doit être coordonnée avec celle du contenant `stopTimeout`. Assurez-vous que `stopTimeout` dépasse le `Grace` période à autoriser Fluent Bit pour terminer le rinçage avant de recevoir `SIGKILL`. Par exemple, si `Grace` est de 120 secondes, réglé `stopTimeout` à 150 secondes.

L'exemple suivant montre un Fluent Bit configuration avec tous les paramètres recommandés pour les scénarios à haut débit :

```
[SERVICE]
    # Flush logs every 1 second
    Flush 1
    # Wait 120 seconds during shutdown to flush remaining logs
    Grace 120
    # Directory for filesystem buffering
    storage.path             /var/log/flb-storage/
    # Limit chunks stored 'up' in memory (reduce for memory-constrained environments)
    storage.max_chunks_up    32
    # Flush backlog chunks to destinations during shutdown (prevents log loss)
    storage.backlog.flush_on_shutdown On
    # Minimum seconds between retries
    scheduler.base           10
    # Maximum seconds between retries (exponential backoff cap)
    scheduler.cap            60

[INPUT]
    Name forward
    unix_path /var/run/fluent.sock
    # Run input in separate thread to prevent blocking
    threaded true
    # Enable filesystem buffering for persistence
    storage.type filesystem

[OUTPUT]
    Name cloudwatch_logs
    Match *
    region us-west-2
    log_group_name /aws/ecs/my-app
    log_stream_name $(ecs_task_id)
    # Use multiple workers for parallel processing
    workers 2
    # Retry failed flushes up to 15 times
    retry_limit 15
    # Maximum disk space for buffered data for this output
    storage.total_limit_size 10G
```

Comprendre les scénarios de perte de données

Les enregistrements peuvent être perdus lors de pannes prolongées ou de problèmes liés aux destinations de sortie. Les recommandations de configuration présentées dans ce guide constituent des approches efficaces pour minimiser les pertes de données, mais ne peuvent garantir l'absence

de perte lors de pannes prolongées. La compréhension de ces scénarios vous permet de configurer Fluent Bit pour maximiser la résilience.

Les enregistrements peuvent être perdus de deux manières : les enregistrements les plus anciens sont supprimés lorsque le stockage est plein, ou les enregistrements les plus récents sont rejetés lorsque le système ne peut pas accepter plus de données.

Les plus vieux records ont été déposés

Les enregistrements mis en mémoire tampon les plus anciens sont supprimés lorsque les nouvelles tentatives sont épuisées ou lorsque `storage.total_limit_size` se remplit et doit faire de la place pour de nouvelles données.

Limite de nouvelles tentatives dépassée

Se produit après AWS le plugin réessaie (si `auto_retry_requests true`) plus 1 initiale Fluent Bit tentative plus `retry_limit` réessaie. Pour atténuer, réglez `retry_limit no_limits` par plugin OUTPUT pour une infinité de tentatives :

```
[OUTPUT]
    Name                        cloudwatch_logs
    Match                       ApplicationLogs
    retry_limit                 no_limits
    auto_retry_requests         true
```

> **Important**
>
> Les tentatives infinies empêchent la perte d'enregistrements en raison de l'épuisement des nouvelles tentatives, mais peuvent provoquer `storage.total_limit_size` à remplir.

Limite de stockage atteinte (mise en mémoire tampon du système de fichiers)

Se produit lorsque la destination de sortie n'est pas disponible plus longtemps que celle que vous avez configurée `storage.total_limit_size` peut être mis en mémoire tampon. Par exemple, une mémoire tampon de 10 Go à 1 MB/s le taux de journalisation fournit environ 2,7 heures de mise en mémoire tampon. Pour atténuer, augmenter `storage.total_limit_size` par plug-in OUTPUT et provisionnez un stockage de tâches Amazon ECS adéquat :

```
[OUTPUT]
    Name                        cloudwatch_logs
    Match                       ApplicationLogs
    storage.total_limit_size    10G
```

Nouveaux enregistrements rejetés

Les enregistrements les plus récents sont supprimés lorsque l'espace disque est épuisé ou lorsque la saisie est suspendue en raison de `Mem_Buf_Limit`.

Espace disque épuisé (mise en mémoire tampon du système de fichiers)

Se produit lorsque l'espace disque est complètement épuisé. Fluent Bit ne met pas les nouveaux enregistrements en file d'attente et ils sont perdus. Pour atténuer, additionnez tous `storage.total_limit_size` valorise et fournit un stockage des tâches Amazon ECS adéquat. Pour plus d'informations, consultez Exigences en matière de stockage des tâches Amazon ECS.

Limite de mémoire atteinte (mise en mémoire tampon)

Se produit lorsque la destination de sortie n'est pas disponible et que la mémoire tampon est pleine. Les plugins de saisie suspendus cessent d'accepter de nouveaux enregistrements. Pour atténuer, utilisez `storage.type filesystem` pour une meilleure résilience, ou une augmentation `Mem_Buf_Limit`.

Meilleures pratiques pour minimiser les pertes de données

Prenez en compte les bonnes pratiques suivantes pour minimiser les pertes de données :

- Utiliser la mise en mémoire tampon du système de fichiers — Coffret `storage.type filesystem` pour une meilleure résilience en cas de panne.
- Dimensionnez le rangement de manière — Calculer `storage.total_limit_size` en fonction du taux de journalisation et de la fenêtre de restauration souhaitée.
- Provisionner un disque adéquat — Assurez-vous que la tâche Amazon ECS dispose d'un stockage éphémère suffisant, Amazon EBS ou Amazon EFS.
- Configurer le comportement des nouvelles tentatives — Équilibre entre `retry_limit` (perd des records après des essais épuisants) et `no_limits` (réessaie indéfiniment mais peut remplir la mémoire).

Utilisez la journalisation multidestination pour plus de fiabilité

L'envoi de journaux vers plusieurs destinations élimine les points de défaillance uniques. Par exemple, si CloudWatch Les journaux subissent une panne, mais les journaux parviennent toujours à Amazon S3.

Multi-destination la journalisation offre les avantages suivants. Le plug-in de sortie Amazon S3 prend également en charge des options de compression telles que le format gzip et Parquet, ce qui permet de réduire les coûts de stockage. Pour plus d'informations, voir Compression S3 dans le Fluent Bit documentation.

Multi-destination la journalisation peut apporter les avantages suivants :

- Redondance — Si une destination échoue, les journaux parviennent toujours à l'autre.
- Rétablissement — Reconstituer les lacunes d'un système par rapport à l'autre.
- Durabilité — Archivez les journaux dans Amazon S3 pour les conserver à long terme.
- Optimisation des coûts — Conservez les journaux récents dans un service de requête rapide comme CloudWatch Journaux avec une conservation plus courte, tout en archivant tous les journaux sur un stockage Amazon S3 à moindre coût pour une conservation à long terme.

Les suivants Fluent Bit la configuration envoie des journaux aux deux CloudWatch Logs et Amazon S3 :

```
[OUTPUT]
    Name cloudwatch_logs
    Match *
    region us-west-2
    log_group_name /aws/ecs/my-app
    log_stream_name $(ecs_task_id)
    workers 2
    retry_limit 15

[OUTPUT]
    Name s3
    Match *
    bucket my-logs-bucket
    region us-west-2
    total_file_size 100M
    s3_key_format /fluent-bit-logs/$(ecs_task_id)/%Y%m%d/%H/%M/$UUID
    upload_timeout 10m
```

```
    # Maximum disk space for buffered data for this output
    storage.total_limit_size 5G
```

Les deux sorties utilisent le même `Match *` modèle, de sorte que tous les enregistrements sont envoyés aux deux destinations indépendamment. Lors d'une panne d'une destination, les journaux continuent de circuler vers l'autre tandis que les vidages échoués s'accumulent dans la mémoire tampon du système de fichiers pour une nouvelle tentative ultérieure.

Utilisez la journalisation basée sur des fichiers avec le plugin Tail Input

Pour les scénarios à haut débit où la perte de journaux constitue un problème critique, vous pouvez utiliser une autre approche : demandez à votre application d'écrire des journaux dans des fichiers sur disque et de configurer Fluent Bit pour les lire à l'aide du `tail` plugin de saisie. Cette approche contourne complètement la couche du pilote de journalisation Docker.

File-based la journalisation avec le plugin tail offre les avantages suivants :

- Suivi des décalages — Le plugin tail peut stocker les décalages de fichiers dans un fichier de base de données (en utilisant le DB en option), offrant une durabilité sur Fluent Bit redémarre. Cela permet d'éviter la perte de journaux lors du redémarrage du conteneur.
- Input-level mise en mémoire tampon — Vous pouvez configurer les limites de mémoire tampon directement sur le plug-in d'entrée en utilisant `Mem_Buf_Limit`, offrant un contrôle plus précis de l'utilisation de la mémoire.
- Évite la surcharge de Docker — Les journaux vont directement d'un fichier à Fluent Bit sans passer par les buffers de journalisation de Docker.

Pour utiliser cette approche, votre application doit écrire des journaux dans des fichiers au lieu de `stdout`. Le contenant de l'application et le Fluent Bit montez en conteneur un volume partagé dans lequel les fichiers journaux sont stockés.

L'exemple suivant montre une configuration d'entrée de queue avec les meilleures pratiques :

```
[INPUT]
    Name tail
    # File path or glob pattern to tail
    Path /var/log/app.log
    # Database file for storing file offsets (enables resuming after restart)
    DB /var/log/flb_tail.db
    # when true, controls that only fluent-bit will access the database (improves
 performance)
```

```
    DB.locking true
    # Skip long lines instead of skipping the entire file
    Skip_Long_Lines On
    # How often (in seconds) to check for new files matching the glob pattern
    Refresh_Interval 10
    # Extra seconds to monitor a file after rotation to account for pending flush
    Rotate_Wait 30
    # Maximum size of the buffer for a single line
    Buffer_Max_Size 10MB
    # Initial allocation size for reading file data
    Buffer_Chunk_Size 1MB
    # Maximum memory buffer size (tail pauses when full)
    Mem_Buf_Limit 75MB
```

Lorsque vous utilisez le plug-in Tail Input, tenez compte des points suivants :

- Implémentez la rotation des journaux de vos applications afin d'éviter l'épuisement des disques. Surveillez les indicateurs de volume sous-jacents pour évaluer les performances.
- Pensez à des paramètres tels que `Ignore_Older`, `Read_from_Head`, et des analyseurs multilignes en fonction du format de votre journal.

Pour plus d'informations, voir Queue dans le Fluent Bit documentation. Pour connaître les meilleures pratiques, voir Configuration de la queue avec les meilleures pratiques dans le AWS pour Fluent Bit guide de dépannage.

Connectez-vous directement à FireLens

Lorsque le pilote de journal `awsfirelens` est spécifié dans une définition de tâche, l'agent de conteneur Amazon ECS injecte les variables d'environnement suivantes dans le conteneur :

`FLUENT_HOST`

L'adresse IP attribuée à FireLens contenant.

> **Note**
>
> Si vous utilisez EC2 avec `bridge` mode réseau, le `FLUENT_HOST` la variable d'environnement de votre conteneur d'applications peut devenir inexacte après un redémarrage du FireLens conteneur log router (le conteneur avec `firelensConfiguration` objet dans sa définition de conteneur). Cela est dû au fait que `FLUENT_HOST` est une adresse IP dynamique qui peut changer après un

redémarrage. La journalisation directe depuis le conteneur de l'application vers l'adresse IP `FLUENT_HOST` peut commencer à échouer après le changement d'adresse. Pour obtenir plus d'informations sur le redémarrage de conteneurs individuels, consultez la section Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`FLUENT_PORT`

Port sur lequel le protocole Fluent Forward écoute.

Vous pouvez utiliser ces variables d'environnement pour vous connecter directement au Fluent Bit connectez le routeur à partir du code de votre application à l'aide du protocole Fluent Forward, au lieu d'écrire dans `stdout`. Cette approche contourne la couche du pilote de journalisation Docker, ce qui présente les avantages suivants :

- Latence plus faible — Les journaux vont directement à Fluent Bit sans passer par l'infrastructure de journalisation de Docker.
- Journalisation structurée — Envoyez des données de journal structurées de manière native sans surcharge d'encodage JSON.
- Meilleur contrôle — Votre application peut implémenter sa propre logique de mise en mémoire tampon et de gestion des erreurs.

Les bibliothèques d'enregistreurs Fluent suivantes prennent en charge le protocole Fluent Forward et peuvent être utilisées pour envoyer des journaux directement à Fluent Bit:

- Va — enregistreur de fluent-golang
- Python — enregistreur de fluidité Python
- Java — enregistreur de fluent-java
- Node.js — nœud fluent-logger
- Rubis — Fluent-Logger-Ruby

Configurer la limite de mémoire tampon de Docker

Lorsque vous créez une définition de tâche, vous pouvez spécifier le nombre de lignes de journal mises en mémoire tampon en spécifiant la valeur dans `log-driver-buffer-limit`. Cela contrôle

la mémoire tampon entre Docker et Fluent Bit. Pour plus d'informations, consultez la section Pilote de journalisation dans la documentation Docker.

Utilisez cette option lorsque le débit est élevé, car Docker risque de manquer de mémoire tampon et de rejeter les messages tampons afin d'en ajouter de nouveaux.

Lorsque vous utilisez cette option, tenez compte des points suivants :

- Cette option est prise en charge sur le type de lancement Amazon EC2 et le type de lancement Fargate avec la version de plateforme `1.4.0` ou une version ultérieure.
- L'option n'est valide que lorsque `logDriver` est défini sur `awsfirelens`.
- La limite par défaut du tampon est de `1048576` lignes de journal.
- La limite de mémoire tampon doit être supérieure ou égale à `0` et inférieure à `536870912` lignes de journal.
- La quantité maximale de mémoire utilisée pour cette mémoire tampon est le produit de la taille de chaque ligne de journal par la taille de la mémoire tampon. Par exemple, si les lignes de journal de l'application sont en moyenne 2 KiB, une limite de mémoire tampon de 4096 utiliserait au maximum 8 MiB. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs, en plus du tampon mémoire du pilote de journalisation.

La définition de tâche suivante montre comment configurer `log-driver-buffer-limit`:

```
{
    "containerDefinitions": [
        {
            "name": "my_service_log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:3",
            "cpu": 0,
            "memoryReservation": 51,
            "essential": true,
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
        {
            "essential": true,
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "name": "app",
```

```
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "firehose",
                    "region": "us-west-2",
                    "delivery_stream": "my-stream",
                    "log-driver-buffer-limit": "52428800"
                }
            },
            "dependsOn": [
                {
                    "containerName": "my_service_log_router",
                    "condition": "START"
                }
            ],
            "memoryReservation": 100
        }
    ]
}
```

AWS for Fluent Bit référentiels d'images pour Amazon ECS

AWS fournit une Fluent Bit image avec plugins pour les deux CloudWatch Bûches et lance à incendie. Nous vous recommandons d'utiliser Fluent Bit comme routeur de journal car il a un taux d'utilisation des ressources inférieur à Fluentd. Pour plus d'informations, voir CloudWatch Journaux pour Fluent Bit et Amazon Kinesis Firehose pour Fluent Bit.

Le AWS pour Fluent Bit l'image est disponible sur Amazon ECR à la fois dans la galerie publique Amazon ECR et dans un référentiel Amazon ECR pour une haute disponibilité.

Galerie publique Amazon ECR

Le AWS pour Fluent Bit L'image est disponible sur la galerie publique Amazon ECR. Il s'agit de l'emplacement recommandé pour télécharger AWS pour Fluent Bit image car il s'agit d'un référentiel public et peut être utilisé par tous Régions AWS. Pour plus d'informations, consultez aws-for-fluent-bit sur la galerie publique Amazon ECR.

Linux

Le AWS pour Fluent Bit l'image de la galerie publique Amazon ECR est compatible avec le système d'exploitation Amazon Linux avec `ARM64` ou `x86-64` architecture.

Vous pouvez tirer AWS pour Fluent Bit image depuis la galerie publique Amazon ECR en spécifiant l'URL du référentiel avec la balise d'image souhaitée. Les étiquettes d'image disponibles peuvent être trouvées sur la page Étiquettes d'image dans la galerie publique Amazon ECR.

L'exemple suivant montre la syntaxe à utiliser pour la CLI Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Par exemple, vous pouvez extraire la dernière image de la famille « 3.x » AWS pour Fluent Bit versions à l'aide de cette commande Docker CLI.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:3
```

> **Note**
>
> Les extractions non authentifiées sont autorisées, mais ont une limite de débit inférieure à celle des extractions authentifiées. Pour vous authentifier à l'aide de votre AWS compte avant de le retirer, utilisez la commande suivante.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
>  AWS --password-stdin public.ecr.aws
> ```

AWS for Fluent Bit 3.0.0

En plus de l'existant AWS pour Fluent Bit versions `2.x`, AWS pour Fluent Bit prend en charge une nouvelle version majeure `3.x`. La nouvelle version majeure inclut la mise à niveau des images d'Amazon Linux 2 vers Amazon Linux 2023 et Fluent Bit version `1.9.10` pour `4.1.1`. Pour plus d'informations, consultez le AWS pour Fluent Bit référentiel sur GitHub.

Les exemples suivants illustrent les balises mises à jour pour AWS pour Fluent Bit `3.x` images :

Vous pouvez utiliser des balises multi-architectures pour AWS pour Fluent Bit image.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:3
```

Windows

Le AWS pour Fluent Bit l'image de la galerie publique Amazon ECR soutient le `AMD64` architecture avec les systèmes d'exploitation suivants :

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Conteneurs Windows qui sont allumés AWS Fargate ne soutient pas FireLens.

Vous pouvez tirer AWS pour Fluent Bit image depuis la galerie publique Amazon ECR en spécifiant l'URL du référentiel avec la balise d'image souhaitée. Les étiquettes d'image disponibles peuvent être trouvées sur la page Étiquettes d'image dans la galerie publique Amazon ECR.

L'exemple suivant montre la syntaxe à utiliser pour la CLI Docker.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:tag
```

Par exemple, vous pouvez extraire la dernière écurie AWS pour Fluent Bit image à l'aide de cette commande Docker CLI.

```
docker pull public.ecr.aws/aws-observability/aws-for-fluent-bit:windowsservercore-
stable
```

> **Note**
>
> Les extractions non authentifiées sont autorisées, mais ont une limite de débit inférieure à celle des extractions authentifiées. Pour vous authentifier à l'aide de votre AWS compte avant de le retirer, utilisez la commande suivante.
>
> ```
> aws ecr-public get-login-password --region us-east-1 | docker login --username
>  AWS --password-stdin public.ecr.aws
> ```

Amazon ECR

Le AWS pour Fluent Bit, l'image est disponible sur Amazon ECR pour une haute disponibilité. Les commandes suivantes peuvent être utilisées pour récupérer les URI des images et établir la disponibilité des images dans un environnement donné Région AWS.

Linux

La dernière écurie AWS pour Fluent Bit, l'URI de l'image peut être récupéré à l'aide de la commande suivante.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/stable \
      --region us-east-1
```

Toutes les versions du AWS pour Fluent Bit, l'image peut être répertoriée à l'aide de la commande suivante pour interroger le paramètre Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit \
      --region us-east-1
```

La plus récente écurie AWS pour Fluent Bit, l'image peut être référencée dans un CloudFormation modèle en faisant référence au nom du magasin de paramètres de Systems Manager. Voici un exemple :

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/stable
```

> **Note**
>
> Si la commande échoue ou s'il n'y a pas de sortie, l'image n'est pas disponible dans Région AWS dans lequel la commande est appelée.

Windows

La dernière écurie AWS pour Fluent Bit, l'URI de l'image peut être récupéré à l'aide de la commande suivante.

```
aws ssm get-parameters \
      --names /aws/service/aws-for-fluent-bit/windowsservercore-stable \
```

```
      --region us-east-1
```

Toutes les versions du AWS pour Fluent Bit, l'image peut être répertoriée à l'aide de la commande suivante pour interroger le paramètre Systems Manager Parameter Store.

```
aws ssm get-parameters-by-path \
      --path /aws/service/aws-for-fluent-bit/windowsservercore \
      --region us-east-1
```

La dernière écurie AWS pour Fluent Bit, l'image peut être référencée dans un CloudFormation modèle en faisant référence au nom du magasin de paramètres de Systems Manager. Voici un exemple :

```
Parameters:
  FireLensImage:
    Description: Fluent Bit image for the FireLens Container
    Type: AWS::SSM::Parameter::Value<String>
    Default: /aws/service/aws-for-fluent-bit/windowsservercore-stable
```

Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens

Pour utiliser le routage des journaux personnalisé avec FireLens, vous devez spécifier les éléments suivants dans votre définition de tâche :

- Conteneur de routeur journal contenant une configuration FireLens. Nous recommandons que le conteneur soit marqué comme `essential`.
- Un ou plusieurs conteneurs d'application contenant une configuration de journal spécifiant le pilote de journal `awsfirelens`.
- Un rôle IAM de tâche Amazon Resource Name (ARN) qui contient les autorisations nécessaires à la tâche pour acheminer les journaux.

Lorsque vous créez une nouvelle définition de tâche à l'aide du Console de gestion AWS, il existe un FireLens section d'intégration qui facilite l'ajout d'un conteneur Log Router. Pour plus d’informations, consultez Création d’une définition de tâche Amazon ECS à l’aide de la console.

Amazon ECS convertit la configuration du journal et génère la configuration de sortie Fluentd ou Fluent Bit. La configuration de sortie est montée dans le conteneur de routage des journaux à `/fluent-bit/etc/fluent-bit.conf` pour Fluent Bit et `/fluentd/etc/fluent.conf` pour Fluentd.

Important

FireLens écoute sur le port `24224`. Par conséquent, afin de garantir que FireLens log router n'est pas accessible en dehors de la tâche, vous ne devez pas autoriser le trafic entrant sur le port `24224` dans le groupe de sécurité utilisé par votre tâche. Pour les tâches utilisant le mode réseau `awsvpc`, il s'agit du groupe de sécurité associé à la tâche. Pour les tâches utilisant le mode réseau `host`, il s'agit du groupe de sécurité associé à l'instance Amazon EC2 qui héberge la tâche. Pour les tâches utilisant le mode réseau `bridge`, ne créez pas de mappages de ports utilisant le port `24224`.

Par défaut, Amazon ECS ajoute des champs supplémentaires dans vos entrées de journal qui aident à identifier la source des journaux.

- `ecs_cluster` : nom du cluster dont la tâche fait partie.
- `ecs_task_arn` : Amazon Resource Name (ARN) de la tâche dont le conteneur fait partie.
- `ecs_task_definition` : nom et révision de la définition de tâche que la tâche utilise.
- `ec2_instance_id` : ID de l'instance Amazon EC2 sur laquelle le conteneur est hébergé. Ce champ est uniquement valide pour les tâches qui utilisent le type de lancement EC2.

Vous pouvez définir `enable-ecs-log-metadata` sur `false` si vous ne souhaitez pas les métadonnées.

L'exemple de définition de tâche suivant définit un conteneur de routeur de journaux qui utilise Fluent Bit pour acheminer ses journaux vers CloudWatch Journaux. Il définit également un conteneur d'application qui utilise une configuration de journal pour acheminer les journaux vers Amazon Data Firehose et définit la mémoire utilisée pour mettre en mémoire tampon les événements à 2 Mio.

Note

Pour plus d'exemples de définitions de tâches, voir Amazon ECS FireLens exemples sur GitHub.

```
{
  "family": "firelens-example-firehose",
  "taskRoleArn": "arn:aws:iam::123456789012:role/ecs_task_iam_role",
```

```
  "containerDefinitions": [
    {
            "name": "log_router",
            "image": "public.ecr.aws/aws-observability/aws-for-fluent-bit:3",
            "cpu": 0,
            "memoryReservation": 51,
            "portMappings": [],
            "essential": true,
            "environment": [],
            "mountPoints": [],
            "volumesFrom": [],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/ecs-aws-firelens-sidecar-container",
                    "mode": "non-blocking",
                    "awslogs-create-group": "true",
                    "max-buffer-size": "25m",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "firelens"
                },
                "secretOptions": []
            },
            "systemControls": [],
            "firelensConfiguration": {
                "type": "fluentbit"
            }
        },
    {
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:latest",
      "name": "app",
      "logConfiguration": {
        "logDriver": "awsfirelens",
        "options": {
          "Name": "firehose",
          "region": "us-west-2",
          "delivery_stream": "my-stream",
          "log-driver-buffer-limit": "1048576"
        }
      },
      "memoryReservation": 100
    }
  ]
```

```
}
```

Les paires clés/valeurs spécifiées en tant qu'options dans l'objet `logConfiguration` sont utilisées pour générer la configuration de sortie Fluentd ou Fluent Bit. Voici un exemple de code à partir d'une définition de sortie Fluent Bit.

```
[OUTPUT]
    Name   firehose
    Match  app-firelens*
    region us-west-2
    delivery_stream my-stream
```

> **Note**
>
> FireLens gère le `match` configuration. Vous ne spécifiez pas la configuration `match` dans votre définition de tâche.

Utilisation d’un fichier de configuration

Vous pouvez spécifier un fichier de configuration personnalisé. Le format du fichier de configuration est le format natif du routeur de journal que vous utilisez. Pour plus d’informations, consultez les sections Syntaxe du fichier de configuration Fluentd et Fichier de configuration YAML.

Dans votre fichier de configuration personnalisé, pour les tâches utilisant le mode réseau `bridge` ou `awsvpc`, ne définissez pas d'entrée de transfert Fluentd ou Fluent Bit sur TCP, car FireLens l'ajoute à la configuration d'entrée.

Votre configuration FireLens doit contenir les options suivantes pour spécifier un fichier de configuration personnalisé :

`config-file-type`

Emplacement source du fichier de configuration personnalisé. Les options disponibles sont `s3` ou `file`.

> **Note**
>
> Tâches hébergées sur AWS Fargate ou les instances gérées Amazon ECS ne prennent en charge que `file` type de fichier de configuration. Vous pouvez toujours utiliser les

fichiers de configuration hébergés dans Amazon S3 avec ces types de lancement. Pour ce faire, utilisez le AWS pour Fluent Bit conteneur d'initialisation. Pour plus d'informations, voir Processus d'initialisation pour Fluent Bit sur ECS, prise en charge de plusieurs configurations sur GitHub.

`config-file-value`

Source du fichier de configuration personnalisé. Si le type de fichier de configuration `s3` est utilisé, la valeur du fichier de configuration est l'ARN complet du compartiment Amazon S3 et du fichier. Si le type de fichier de configuration `file` est utilisé, la valeur du fichier de configuration est le chemin d'accès complet du fichier de configuration qui existe soit dans l'image du conteneur, soit sur un volume monté dans le conteneur.

Important

Lorsque vous utilisez un fichier de configuration personnalisé, vous devez spécifier un chemin différent de celui utilisé par FireLens. Amazon ECS réserve les chemins de fichier `/fluent-bit/etc/fluent-bit.conf` pour Fluent Bit et `/fluentd/etc/fluent.conf` pour Fluentd.

L'exemple suivant montre la syntaxe requise lors de la spécification d'une configuration personnalisée.

Important

Pour spécifier un fichier de configuration personnalisé hébergé dans Amazon S3, vérifiez que vous avez créé un rôle IAM d'exécution de tâche avec les autorisations appropriées.

Voici la syntaxe requise lors de la spécification d'une configuration personnalisée.

```
{
  "containerDefinitions": [
    {
      "essential": true,
      "image": "906394416424.dkr.ecr.us-west-2.amazonaws.com/aws-for-fluent-bit:3",
```

```
      "name": "log_router",
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "config-file-type": "s3 | file",
          "config-file-value": "arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf
 | filepath"
        }
      }
    }
  ]
}
```

> **Note**
>
> Tâches hébergées sur AWS Fargate ou les instances gérées Amazon ECS ne prennent en charge que `file` type de fichier de configuration. Vous pouvez toujours utiliser les fichiers de configuration hébergés dans Amazon S3 avec ces types de lancement. Pour ce faire, utilisez le AWS pour Fluent Bit conteneur d'initialisation. Pour plus d'informations, voir Processus d'initialisation pour Fluent Bit sur ECS, prise en charge de plusieurs configurations sur GitHub.

En utilisant des produits non AWS images de conteneurs dans Amazon ECS

Utilisez un registre privé pour stocker vos informations d'identification AWS Secrets Manager, puis référencez-les dans votre définition de tâche. Cela permet de référencer des images de conteneurs qui existent dans des registres privés en dehors de AWS qui nécessitent une authentification dans vos définitions de tâches. Cette fonction est prise en charge par les tâches hébergées sur Fargate, les instances Amazon EC2 et les instances externes utilisant Amazon ECS Anywhere.

> **Important**
>
> Si votre définition de tâche fait référence à une image stockée dans Amazon ECR, cette rubrique ne s'applique pas. Pour plus d'informations, consultez Utilisation d'images Amazon ECR avec Amazon ECS dans le Guide de l'utilisateur Amazon Elastic Container Registry.

Pour les tâches hébergées sur des instances Amazon EC2, cette fonction exige nécessite la version `1.19.0` ou ultérieure de l'agent de conteneur. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour les tâches hébergées sur Fargate, cette fonction nécessite une version `1.2.0` de la plateforme ou une version ultérieure. Pour plus d'informations, veuillez consulter Versions de plateforme Fargate pour Amazon ECS.

Dans votre définition du conteneur, spécifiez l'objet `repositoryCredentials` avec les détails du secret que vous avez créé. Le secret référencé peut provenir d'un autre Région AWS ou un compte différent de celui de la tâche qui l'utilise.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS, AWS CLI, ou AWS SDK, si le secret existe dans le même Région AWS comme tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Si le secret existe dans un autre compte, l'ARN complet du secret doit être spécifié. Lors de l'utilisation du Console de gestion AWS, l'ARN complet du secret doit toujours être spécifié.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

Remplacez les paramètres suivants :

- *`private-repo`* avec le nom d'hôte du référentiel privé
- *`private-image`* avec le nom de l'image
- *`arn:aws:secretsmanager:region:aws_account_id:secret:secret_name`* avec le nom de ressource Amazon (ARN) secret

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
```

```
        }
    }
]
```

> **Note**
>
> Une autre méthode d'activation de l'authentification de registre privé utilise des variables d'environnement d'agent de conteneur Amazon ECS pour s'authentifier auprès de registres privés. Cette méthode est prise en charge uniquement pour les tâches hébergées sur des instances Amazon EC2. Pour plus d'informations, consultez Configuration des instances de conteneur Amazon ECS pour les images Docker privées.

Pour utiliser un registre privé

1. La définition de tâche doit comporter un rôle d'exécution de tâche. Cela permet à l'agent de conteneur d'extraire l'image de conteneur. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

   L'authentification de registre privé permet à vos tâches Amazon ECS d'extraire des images de conteneurs depuis des registres privés extérieurs à AWS (tels que Docker Hub, Quay.io, ou votre propre registre privé) qui nécessitent des informations d'authentification. Cette fonctionnalité utilise Secrets Manager pour stocker en toute sécurité vos informations d'identification de registre, qui sont ensuite référencées dans votre définition de tâche à l'aide du paramètre `repositoryCredentials`.

   Pour plus d'informations sur la configuration de l'authentification dans le registre privé, voir En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

   Pour donner accès aux secrets contenant vos informations d'identification du registre privé, ajoutez les autorisations suivantes en tant que politique intégrée au rôle d'exécution des tâches. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

   - `secretsmanager:GetSecretValue` : nécessaire pour récupérer les informations d'identification du registre privé auprès de Secrets Manager.
   - `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé KMS personnalisée et non la clé par défaut. L'Amazon Resource Name (ARN) de votre clé personnalisée doit être ajouté en tant que ressource.

Voici un exemple de stratégie en ligne qui ajoute les autorisations.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secret_name",
                "arn:aws:kms:us-east-1:111122223333:key/key_id"
            ]
        }
    ]
}
```

2. Utiliser AWS Secrets Manager pour créer un secret pour les informations d'identification de votre registre privé. Pour plus d’informations sur la création d’un secret, consultez la section Création d’un secret AWS Secrets Manager dans le Guide de l’utilisateur AWS Secrets Manager .

   Saisissez vos informations d’identification de registre privé en utilisant le format suivant :

   ```
   {
     "username" : "privateRegistryUsername",
     "password" : "privateRegistryPassword"
   }
   ```

3. Enregistrer une définition de tâche. Pour plus d’informations, consultez the section called “Création d'une définition de tâche à l'aide de la console”.

Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs

Vous pouvez activer une politique de redémarrage pour chaque conteneur essentiel et non essentiel défini dans votre définition de tâche, afin de surmonter plus rapidement les défaillances transitoires et de maintenir la disponibilité des tâches. Lorsque vous activez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur s'il existe, sans avoir à remplacer la tâche.

Les politiques de redémarrage ne sont pas activées par défaut pour les conteneurs. Lorsque vous activez une politique de redémarrage pour un conteneur, vous pouvez spécifier les codes de sortie pour lesquels le conteneur ne sera pas redémarré. Il peut s'agir de codes de sortie indiquant le succès, tels que des codes de sortie `0`, qui ne nécessitent pas de redémarrage. Vous pouvez également spécifier la durée pendant laquelle un conteneur doit fonctionner correctement avant qu'un redémarrage puisse être tenté. Pour obtenir plus d'informations sur ces paramètres, consultez Politique de redémarrage. Pour un exemple de définition de tâche qui spécifie ces valeurs, consultez la section Spécification d'une politique de redémarrage de conteneur dans une définition de tâche Amazon ECS.

Vous pouvez utiliser le point de terminaison des métadonnées des tâches Amazon ECS ou CloudWatch Container Insights permet de surveiller le nombre de fois qu'un conteneur a redémarré. Pour plus d'informations sur le point de terminaison des métadonnées de tâche, consultez les sections Point de terminaison des métadonnées de tâches Amazon ECS version 4 et Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate. Pour plus d'informations sur les métriques de Container Insights pour Amazon ECS, consultez Statistiques d'Amazon ECS Container Insights dans le Amazon CloudWatch Guide de l'utilisateur.

Les politiques de redémarrage des conteneurs sont prises en charge par les tâches hébergées sur Fargate, les instances Amazon EC2 et les instances externes utilisant Amazon ECS Anywhere.

Considérations

Tenez compte des points suivants avant d'activer une politique de redémarrage pour votre conteneur :

- Les politiques de redémarrage ne sont pas prises en charge pour les conteneurs Windows sur Fargate.
- Pour les tâches hébergées sur des instances Amazon EC2, cette fonction exige nécessite la version `1.86.0` ou ultérieure de l'agent de conteneur. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de

la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

- Si vous utilisez EC2 avec `bridge` mode réseau, `FLUENT_HOST` la variable d'environnement de votre conteneur d'applications peut devenir inexacte après un redémarrage du FireLens conteneur log router (le conteneur avec `firelensConfiguration` objet dans sa définition de conteneur). Cela est dû au fait que `FLUENT_HOST` est une adresse IP dynamique qui peut changer après un redémarrage. La journalisation directe depuis le conteneur de l'application vers l'adresse IP `FLUENT_HOST` peut commencer à échouer après le changement d'adresse. Pour plus d'informations sur `FLUENT_HOST`, consultez Configuration des journaux Amazon ECS pour un débit élevé.
- L'agent Amazon ECS gère les politiques de redémarrage des conteneurs. Si, pour une raison inattendue, l'agent Amazon ECS échoue ou ne fonctionne plus, le conteneur ne sera pas redémarré.
- La période de tentative de redémarrage définie dans votre politique détermine la période (en secondes) pendant laquelle le conteneur doit fonctionner avant qu'Amazon ECS ne redémarre un conteneur.

Spécification d'une politique de redémarrage de conteneur dans une définition de tâche Amazon ECS

Pour spécifier une politique de redémarrage pour un conteneur dans une définition de tâche, dans la définition du conteneur, spécifiez l'objet `restartPolicy`. Pour plus d'informations sur l'objet `restartPolicy`, consultez la section Politique de redémarrage.

Ce qui suit est une définition de tâche utilisant les conteneurs Linux sur Fargate qui configure un serveur Web. La définition du conteneur inclut l'objet `restartPolicy`, avec `enabled` défini sur true pour activer une politique de redémarrage pour le conteneur. Le conteneur doit fonctionner pendant 180 secondes avant de pouvoir être redémarré et ne sera pas redémarré s'il se ferme avec le code de sortie `0`, qui indique un succès.

```
{
  "containerDefinitions": [
    {
      "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\""
      ],
      "entryPoint": ["sh", "-c"],
      "essential": true,
      "image": "public.ecr.aws/docker/library/httpd:2.4",
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fargate-task-definition",
          "awslogs-region": "us-east-1",
          "awslogs-stream-prefix": "ecs"
        }
      },
      "name": "sample-fargate-app",
      "portMappings": [
        {
          "containerPort": 80,
          "hostPort": 80,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      }
    }
  ],
  "cpu": "256",
  "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
  "family": "fargate-task-definition",
  "memory": "512",
  "networkMode": "awsvpc",
  "runtimePlatform": {
    "operatingSystemFamily": "LINUX"
  },
  "requiresCompatibilities": ["FARGATE"]
}
```

Après avoir enregistré une définition de tâche avec l'objet `restartPolicy` dans une définition de conteneur, vous pouvez exécuter une tâche ou créer un service avec cette définition de tâche. Pour

plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS et Création d'un déploiement de mise à jour propagée Amazon ECS.

Transmission de données sensibles vers un conteneur Amazon ECS

Vous pouvez transmettre en toute sécurité des données sensibles, telles que des informations d'identification vers une base de données, dans votre conteneur.

Les secrets, tels que les clés d'API et les informations d'identification de base de données, sont fréquemment utilisés par les applications pour accéder à d'autres systèmes. Ils se composent souvent d'un nom d'utilisateur et d'un mot de passe, d'un certificat ou d'une clé d'API. L'accès à ces secrets doit être limité à des principaux IAM spécifiques qui utilisent IAM et injectés dans des conteneurs lors de l'exécution.

Les secrets peuvent être facilement injectés dans des contenants à partir de AWS Secrets Manager et Amazon EC2 Systems Manager Parameter Store. Ces secrets peuvent être référencés dans votre tâche de l'une des manières suivantes.

1. Ils sont référencés en tant que variables d'environnement qui utilisent le paramètre de définition du conteneur `secrets`.
2. Ils sont référencés comme `secretOptions` si votre plateforme de journalisation nécessite une authentification. Pour plus d'informations, veuillez consulter Options de configuration de la journalisation (langue française non garantie).
3. Ils sont référencés comme des secrets extraits par des images qui utilisent le paramètre de définition de conteneur `repositoryCredentials` si le registre d'où le conteneur est extrait nécessite une authentification. Utilisez cette méthode lors de l'extraction d'images depuis la Galerie publique Amazon ECR. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez la gestion des secrets.

Utilisation AWS Secrets Manager or AWS Systems Manager Parameter Store pour stocker des matériaux secrets

Vous devez stocker en toute sécurité les clés d'API, les informations d'identification de base de données et les autres éléments secrets dans Secrets Manager ou sous forme de paramètre chiffré

dans le magasin de paramètres Systems Manager. Ces services sont similaires car il s'agit tous deux de magasins à valeur clé gérés qui utilisent AWS KMS pour crypter les données sensibles. Secrets Manager, cependant, inclut également la possibilité de faire pivoter automatiquement les secrets, de générer des secrets aléatoires et de partager des secrets entre les comptes. Pour utiliser ces fonctionnalités, utilisez Secrets Manager. Sinon, utilisez des paramètres chiffrés dans le magasin de paramètres Systems Manager.

> **Important**
>
> Si votre secret change, vous devez forcer un nouveau déploiement ou lancer une nouvelle tâche pour récupérer la dernière valeur secrète. Pour plus d'informations, consultez les rubriques suivantes :
>
> - Tâches : arrêtez la tâche, puis démarrez-la. Pour plus d'informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.
> - Service : mettez à jour le service et utilisez l'option « Forcer un nouveau déploiement ». Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Récupération des données d'un compartiment Amazon S3 chiffré

Vous devez stocker les secrets dans un compartiment chiffré Amazon S3 et utiliser des rôles de tâche pour restreindre l'accès à ces secrets. Cela empêche les valeurs des variables d'environnement de s'échapper par inadvertance dans les journaux et d'être révélées lors de l'exécution de `docker inspect`. Dans ce cas, votre application doit être écrite pour lire le secret du compartiment Amazon S3. Pour obtenir des instructions, veuillez consulter Définition du comportement de chiffrement côté serveur par défaut pour les compartiments Amazon S3.

Montez le secret sur un volume à l'aide d'un conteneur sidecar

En raison du risque élevé de fuite de données liées aux variables d'environnement, vous devez exécuter un conteneur annexe qui lit vos secrets depuis AWS Secrets Manager et écrivez-les dans un volume partagé. Ce conteneur peut s'exécuter et sortir avant le conteneur d'applications en utilisant le système de commande de conteneurs Amazon ECS. Ainsi, le conteneur de l'application monte ensuite le volume dans lequel le secret a été écrit. À l'instar de la méthode du compartiment Amazon S3, votre application doit être écrite pour lire le secret du volume partagé. Le volume étant limité à la tâche, il est automatiquement supprimé après l'arrêt de la tâche. Pour un exemple, consultez le projet task-def.json.

Sur Amazon EC2, le volume sur lequel le secret est écrit peut être chiffré à l'aide d'une clé AWS KMS gérée par le client. Sur AWS Fargate, le stockage en volume est automatiquement chiffré à l'aide d'une clé gérée par les services.

Transmission d'une variable d'environnement individuelle à un conteneur Amazon ECS

> **Important**
>
> Nous vous recommandons de stocker vos données sensibles dans AWS Secrets Manager secrets ou AWS Systems Manager Paramètre Stockez les paramètres. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.
>
> Les variables d'environnement spécifiées dans la définition de tâche sont lisibles par tous les utilisateurs et rôles autorisés à effectuer l'action `DescribeTaskDefinition` pour la définition de tâche.

Vous pouvez transmettre des variables d'environnement à vos conteneurs des manières suivantes :

- Individuellement à l'aide du paramètre de définition de conteneur `environment`. Cela correspond à l'option `--env` de docker container run.
- En bloc, en utilisant le paramètre de définition de conteneur `environmentFiles` pour répertorier un ou plusieurs fichiers contenant les variables d'environnement. Le fichier doit être hébergé dans Amazon S3. Cela correspond à l'option `--env-file` de docker run.

Voici un extrait d'une définition de tâche montrant comment spécifier des variables d'environnement individuelles.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environment": [
                {
                    "name": "variable",
                    "value": "value"
```

```
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmission de variables d'environnement à un conteneur Amazon ECS

> **Important**
>
> Nous vous recommandons de stocker vos données sensibles dans AWS Secrets Manager secrets ou AWS Systems Manager Paramètre Stockez les paramètres. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.
>
> Les fichiers de variables d'environnement sont des objets dans Simple Storage Service (Amazon S3) et toutes les considérations de sécurité Simple Storage Service (Amazon S3) s'appliquent.
>
> Vous ne pouvez pas utiliser le paramètre `environmentFiles` sur les conteneurs Windows et les conteneurs Windows sur Fargate.

Vous pouvez créer un fichier de variables d'environnement et le stocker dans Amazon S3 pour transmettre des variables d'environnement à votre conteneur.

En spécifiant des variables d'environnement dans un fichier, vous pouvez injecter en bloc des variables d'environnement. Dans votre définition de conteneur, spécifiez l'objet `environmentFiles` avec une liste de compartiments Amazon S3 contenant vos fichiers de variables d'environnement.

Amazon ECS n'impose pas de limite de taille pour les variables d'environnement, mais un fichier de variables d'environnement volumineux peut saturer l'espace disque. Chaque tâche qui utilise un fichier de variables d'environnement entraîne le téléchargement d'une copie du fichier sur le disque. Amazon ECS supprime le fichier dans le cadre du nettoyage des tâches.

Pour plus d'informations sur les variables d'environnement prises en charge, veuillez consulter Paramètres de définition de conteneur avancés - Environnement.

Tenez compte des éléments suivants lors de la spécification d'un fichier de variable d'environnement dans une définition de conteneur.

- Pour toute tâche Amazon ECS hébergé sur Amazon EC2, vos instances de conteneur nécessitent une version `1.39.0` ou ultérieure de l'agent de conteneur pour utiliser cette fonction. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- Pour les tâches Amazon ECS sur AWS Fargate, vos tâches doivent utiliser la version de la plateforme `1.4.0` ou version ultérieure (Linux) pour utiliser cette fonctionnalité. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.

  Vérifiez que la variable est prise en charge par la plateforme du système d'exploitation. Pour plus d'informations, consultez the section called “Définitions de conteneur” et the section called “Autres paramètres de définition de tâche”.
- Le fichier doit utiliser `.env` extension de fichier et UTF-8 encodage.
- Le rôle d'exécution de tâche est requis pour utiliser cette fonctionnalité avec les autorisations supplémentaires pour Amazon S3. Cela permet à l'agent de conteneur d'extraire le fichier de variable d'environnement à partir d'Amazon S3. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.
- Il y a une limite de 10 fichiers par définition de tâche.
- Chaque ligne d'un fichier d'environnement doit contenir une variable d'environnement au format `VARIABLE=VALUE`. Des espaces ou des guillemets sont inclus dans les valeurs pour les fichiers Amazon ECS. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées. Pour de plus amples informations sur la syntaxe du fichier de variable d'environnement, consultez la section Définition des variables d'environnement (-e, --env, --env-file) dans la documentation Docker.

  Voici la syntaxe appropriée.

  ```
  #This is a comment and will be ignored
  VARIABLE=VALUE
  ENVIRONMENT=PRODUCTION
  ```
- Si des variables d'environnement sont spécifiées à l'aide du paramètre `environment` dans une définition de conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement.
- Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités par ordre d'entrée. Cela signifie que la première valeur de la variable est utilisée et que les valeurs suivantes des variables dupliquées sont ignorées. Nous vous recommandons d'utiliser des noms de variables uniques.

- Si un fichier d'environnement est spécifié en tant que remplacement de conteneur, il est utilisé. De plus, tous les autres fichiers d'environnement spécifiés dans la définition du conteneur sont ignorés.
- Les règles suivantes s'appliquent à Fargate :
  - Le fichier est traité de manière similaire à un fichier env Docker natif.
  - Les définitions de conteneur qui font référence à des variables d'environnement vides et stockées dans Amazon S3 n'apparaissent pas dans le conteneur.
  - La gestion de l'échappement dans shell n'est pas prise en charge.
  - Le point d'entrée du conteneur interprète les valeurs `VARIABLE`.

Exemple

Voici un extrait d'une définition de tâche indiquant comment spécifier un fichier de variable d'environnement.

```
{
    "family": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            ...
            "environmentFiles": [
                {
                    "value": "arn:aws:s3:::amzn-s3-demo-
bucket/envfile_object_name.env",
                    "type": "s3"
                }
            ],
            ...
        }
    ],
    ...
}
```

Transmission des secrets Secrets Manager par programmation dans Amazon ECS

Au lieu de coder en dur les informations sensibles en texte brut dans votre application, vous pouvez utiliser Secrets Manager pour stocker les données sensibles.

Nous recommandons cette méthode pour récupérer les données sensibles, car si le secret de Secrets Manager est ultérieurement mis à jour, l'application récupère automatiquement la dernière version de celui-ci.

Créez un secret dans Secrets Manager. Après avoir créé un secret dans Secrets Manager, mettez à jour le code de votre application pour récupérer le secret.

Prenez en compte les points suivants avant de sécuriser des données sensibles dans Secrets Manager.

- Seuls les secrets qui stockent des données textuelles, qui sont des secrets créés avec `SecretString` paramètre du CreateSecret API, sont pris en charge. Secrets qui stockent des données binaires, qui sont des secrets créés avec `SecretBinary` paramètre du CreateSecret Les API ne sont pas prises en charge.
- Utilisez les points de terminaison VPC de l'interface pour améliorer les contrôles de sécurité. Vous devez créer les points de terminaison d'un VPC pour Secrets Manager. Pour plus d'informations sur le point de terminaison d'un VPC, veuillez consulter Créer des points de terminaison d'un VPC dans le Guide de l'utilisateur AWS Secrets Manager .
- Le VPC que votre tâche utilise doit utiliser la résolution DNS.
- Votre définition de tâche doit utiliser un rôle de tâche doté des autorisations supplémentaires pour Secrets Manager. Pour plus d’informations, consultez rôle IAM de tâche Amazon ECS.

Créer le secret Secrets Manager

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Pour plus d'informations sur la création de secrets, consultez la rubrique Création d'un secret AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Mettez à jour votre application pour récupérer le secret Secrets Manager par programme

Vous pouvez récupérer des secrets en appelant les API de Secrets Manager directement depuis votre application. Pour plus d’informations, consultez la section Récupérer les secrets depuis AWS Secrets Manager dans le Guide de l’utilisateur AWS Secrets Manager .

Pour récupérer les données sensibles stockées dans AWS Secrets Manager, voir Exemples de code pour AWS Secrets Manager en utilisant AWS SDK dans le AWS Exemples de code SDK Bibliothèque de codes.

Transmission des secrets du magasin de paramètres Systems Manager par programmation dans Amazon ECS

Systems Manager Parameter Store offre un stockage et une gestion sécurisés des secrets. Vous pouvez stocker des données telles que des mots de passe, des chaînes de base de données, des identifiants d'instance EC2 et AMI, ainsi que des codes de licence sous forme de valeurs de paramètres, au lieu d'intégrer ces informations de manière statique dans votre application. Vous pouvez stocker ces valeurs sous forme de texte brut ou de données chiffrées.

Nous recommandons cette méthode de récupération des données sensibles, car si le magasin de paramètres Systems Manager est mis à jour ultérieurement, l'application récupère automatiquement la dernière version.

Prenez en compte les points suivants avant de sécuriser des données sensibles dans Systems Manager Parameter Store.

- Seuls les secrets qui stockent des données texte sont pris en charge. Les secrets qui stockent des données binaires ne sont pas pris en charge.
- Utilisez les points de terminaison d'un VPC de l'interface pour améliorer les contrôles de sécurité.
- Le VPC que votre tâche utilise doit utiliser la résolution DNS.
- Pour les tâches qui utilisent EC2, vous devez utiliser la variable de configuration `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true` de l'agent ECS pour utiliser cette fonctionnalité. Vous pouvez l'ajouter au fichier `/etc/ecs/ecs.config` lors de la création de l'instance de conteneur, ou vous pouvez l'ajouter à une instance existante, puis redémarrer l'agent ECS. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.
- Votre définition de tâche doit utiliser un rôle de tâche doté des autorisations supplémentaires pour le magasin de paramètres Systems Manager. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Créer le paramètre

Vous pouvez utiliser la console System Manager pour créer un paramètre Systems Manager Parameter Store pour vos données sensibles. Pour plus d'informations, consultez Créer un paramètre Systems Manager (console) ou Créer un paramètre Systems Manager (AWS CLI) dans le Guide de l'utilisateur AWS Systems Manager .

Mettez à jour votre application pour récupérer par programme les secrets Systems Manager Parameter Store.

Pour récupérer les données sensibles stockées dans le paramètre Systems Manager Parameter Store, voir Exemples de code pour Systems Manager utilisant AWS SDK dans le AWS Exemples de code SDK Bibliothèque de codes.

Transmission des secrets Secrets Manager via les variables d'environnement Amazon ECS

Lorsque vous injectez un secret en tant que variable d'environnement, vous pouvez spécifier le contenu complet d'un secret ou une clé JSON spécifique au sein d'un secret. Cela vous aide à contrôler les données sensibles exposées à votre conteneur. Pour plus d'informations sur la gestion des versions de secrets, consultez la section Que contient un secret Secrets Manager ? dans le Guide de l'utilisateur AWS Secrets Manager .

Les informations suivantes doivent être prises en compte lors de l'utilisation d'une variable d'environnement pour injecter un secret Secrets Manager dans un conteneur.

- Les données sensibles sont injectées dans votre conteneur lors du démarrage initial du conteneur. Si le secret est ensuite mis à jour ou fait l'objet d'une rotation, le conteneur ne reçoit pas la valeur mise à jour automatiquement. Vous devez lancer une nouvelle tâche ou, si votre tâche fait partie d'un service, vous pouvez mettre à jour le service et utiliser l'option Force new deployment (Forcer un nouveau déploiement) pour forcer le service à lancer une nouvelle tâche.
- Les applications qui s'exécutent sur le conteneur, les journaux du conteneur et les outils de débogage ont accès aux variables d'environnement.
- Pour les tâches Amazon ECS sur AWS Fargate, considérez les points suivants :
  - Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.3.0` ou ultérieure de la plateforme. Pour plus d'informations, veuillez consulter Versions de plateforme Fargate pour Amazon ECS.
  - Pour injecter une clé JSON spécifique ou une version d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows). Pour plus d'informations, veuillez consulter Versions de plateforme Fargate pour Amazon ECS.
- Pour les tâches Amazon ECS sur EC2, les informations suivantes doivent être prises en compte :
  - Pour injecter un secret à l'aide d'une clé JSON spécifique ou d'une version d'un secret, votre instance de conteneur doit avoir la version `1.37.0` ou ultérieure de l'agent de conteneur. Nous

vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou pour injecter un secret dans une configuration de journal, votre instance de conteneur doit avoir la version `1.22.0` ou ultérieure de l'agent de conteneur.

- Utilisez les points de terminaison de VPC d'interface pour renforcer les contrôles de sécurité et pour vous connecter à Secrets Manager via un sous-réseau privé. Vous devez créer les points de terminaison d'un VPC pour Secrets Manager. Pour plus d'informations sur le point de terminaison d'un VPC, veuillez consulter Créer des points de terminaison d'un VPC dans le Guide de l'utilisateur AWS Secrets Manager . Pour plus d'informations sur l'utilisation de Secrets Manager et Amazon VPC, consultez la section Comment se connecter au service Secrets Manager au sein d'un Amazon VPC.
- Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

- Votre définition de tâche doit utiliser un rôle d'exécution de tâche doté des autorisations supplémentaires pour Secrets Manager. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Créez le AWS Secrets Manager secret

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Pour plus d'informations, voir Créez un AWS Secrets Manager secret dans le AWS Secrets Manager Guide de l'utilisateur.

Ajoutez la variable d'environnement à la définition du conteneur

Dans votre définition de conteneur, vous pouvez spécifier les éléments suivants :

- Objet `secrets` contenant le nom de la variable d'environnement à définir dans le conteneur
- Amazon Resource Name (ARN) du secret Secrets Manager
- Paramètres supplémentaires contenant les données sensibles à présenter au conteneur

L'exemple suivant montre la syntaxe complète qui doit être spécifiée pour le secret Secrets Manager.

```
arn:aws:secretsmanager:region:aws_account_id:secret:secret-name:json-key:version-stage:version-id
```

La section suivante décrit les paramètres supplémentaires. Ces paramètres sont facultatifs, mais si vous ne les utilisez pas, vous devez inclure les deux-points `:` pour utiliser les valeurs par défaut. Des exemples sont donnés ci-dessous pour plus de contexte.

`json-key`

Spécifie le nom de la clé dans une paire clé-valeur avec la valeur que vous souhaitez définir comme valeur de variable d'environnement. Seules les valeurs au format JSON sont prises en charge. Si vous ne spécifiez pas de clé JSON, le contenu complet du secret est utilisé.

`version-stage`

Spécifie l'étiquette intermédiaire de la version d'un secret que vous souhaitez utiliser. Si une étiquette intermédiaire de version est spécifiée, vous ne pouvez pas spécifier d'ID de version. Si aucune étape de version n'est spécifiée, le comportement par défaut consiste à récupérer le secret avec l'étiquette `AWSCURRENT` intermédiaire.

Les étiquettes intermédiaires sont utilisées pour suivre les différentes versions d'un secret lorsqu'elles sont mises à jour ou font l'objet d'une rotation. Chaque version d'un secret a une ou plusieurs étiquettes intermédiaires et un ID.

`version-id`

Spécifie l'identifiant unique de la version du secret que vous souhaitez utiliser. Si un ID de version est spécifié, vous ne pouvez pas spécifier d'étiquette intermédiaire de version. Si aucun ID de version n'est spécifié, le comportement par défaut consiste à récupérer le secret avec l'étiquette `AWSCURRENT` intermédiaire.

Les ID de version sont utilisés pour suivre les différentes versions d'un secret lorsqu'elles sont mises à jour ou font l'objet d'une rotation. Chaque version d'un secret a un ID. Pour plus

d'informations, veuillez consulter la rubrique Concepts et termes clés pour AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Exemples de définitions de conteneur

Les exemples suivants montrent comment vous pouvez référencer des secrets Secrets Manager dans vos définitions de conteneur.

Example référencement d'un secret complet

Voici un extrait d'une définition de tâche montrant le format lorsque vous référencez le texte complet d'un secret Secrets Manager.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
    }]
  }]
}
```

Pour accéder à la valeur de ce secret depuis le conteneur, vous devez appeler `$environment_variable_name`.

Example référencement de secrets complets

Voici l'extrait de code d'une définition de tâche présentant le format utilisé pour référencer le texte intégral de plusieurs secrets Secrets Manager.

```
{
  "containerDefinitions": [{
     "secrets": [
      {
        "name": "environment_variable_name1",
         "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      },
      {
        "name": "environment_variable_name2",
```

```
          "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
abcdef"
      },
      {
        "name": "environment_variable_name3",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
ABCDEF"
      }
    ]
  }]
}
```

Pour accéder à la valeur de ce secret à partir du conteneur, vous devez appeler `$environment_variable_name1`, `$environment_variable_name2` et `$environment_variable_name3`.

Example référencement d'une clé spécifique dans un secret

Voici un exemple de sortie d'une commande get-secret-value qui affiche le contenu d'un secret ainsi que l'étiquette intermédiaire de version et l'ID de version qui lui sont associés.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "VersionId": "871d9eca-18aa-46a9-8785-981ddEXAMPLE",
    "SecretString": "{\"username1\":\"password1\",\"username2\":\"password2\",
\"username3\":\"password3\"}",
    "VersionStages": [
        "AWSCURRENT"
    ],
    "CreatedDate": 1581968848.921
}
```

Référence d'une clé spécifique de la sortie précédente dans une définition de conteneur en spécifiant le nom de la clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::"
```

```
    }]
  }]
}
```

Example référencement d'une version secrète spécifique

Voici un exemple de sortie d'une commande describe-secret qui affiche le contenu non chiffré d'un secret ainsi que les métadonnées de toutes les versions du secret.

```
{
    "ARN": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-AbCdEf",
    "Name": "appauthexample",
    "Description": "Example of a secret containing application authorization data.",
    "RotationEnabled": false,
    "LastChangedDate": 1581968848.926,
    "LastAccessedDate": 1581897600.0,
    "Tags": [],
    "VersionIdsToStages": {
        "871d9eca-18aa-46a9-8785-981ddEXAMPLE": [
            "AWSCURRENT"
        ],
        "9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE": [
            "AWSPREVIOUS"
        ]
    }
}
```

Référence d'une étiquette intermédiaire de version spécifique à partir de la sortie précédente dans une définition de conteneur en spécifiant le nom de la clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf::AWSPREVIOUS:"
    }]
  }]
}
```

Référence un ID de version spécifique de la sortie précédente dans une définition de conteneur en spécifiant le nom de clé à la fin de l'ARN.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Example référencement d'une clé spécifique et d'une étiquette intermédiaire de version d'un secret

Ce qui suit montre comment référencer à la fois une clé spécifique dans une étiquette secrète et une étiquette de mise en scène de version spécifique.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1:AWSPREVIOUS:"
    }]
  }]
}
```

Pour spécifier une clé et un ID de version spécifiques, utilisez la syntaxe suivante.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:appauthexample-
AbCdEf:username1::9d4cb84b-ad69-40c0-a0ab-cead3EXAMPLE"
    }]
  }]
}
```

Pour plus d’informations sur la création d’une définition de tâche avec le secret spécifié dans une variable d’environnement, consultez la section Création d’une définition de tâche Amazon ECS à l’aide de la console.

Transmission des paramètres System Manager via les variables d’environnement Amazon ECS

Amazon ECS vous permet d'injecter des données sensibles dans vos conteneurs en stockant vos données sensibles dans AWS Systems Manager Paramètre Stockez les paramètres, puis référencez-les dans la définition de votre conteneur.

Tenez compte des éléments suivants lorsque vous utilisez une variable d’environnement pour injecter un secret Systems Manager dans un conteneur.

- Les données sensibles sont injectées dans votre conteneur lors du démarrage initial du conteneur. Si le secret est ensuite mis à jour ou fait l'objet d'une rotation, le conteneur ne reçoit pas la valeur mise à jour automatiquement. Vous devez lancer une nouvelle tâche ou, si votre tâche fait partie d'un service, vous pouvez mettre à jour le service et utiliser l'option Force new deployment (Forcer un nouveau déploiement) pour forcer le service à lancer une nouvelle tâche.
- Pour les tâches Amazon ECS sur AWS Fargate, les éléments suivants doivent être pris en compte :
  - Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.3.0` ou ultérieure de la plateforme. Pour plus d'informations, veuillez consulter Versions de plateforme Fargate pour Amazon ECS.
  - Pour injecter une clé JSON spécifique ou une version d'un secret en tant que variable d'environnement ou dans une configuration de journal, vous devez utiliser la version `1.4.0` de la plateforme ou une version ultérieure (Linux) ou `1.0.0` (Windows). Pour plus d'informations, veuillez consulter Versions de plateforme Fargate pour Amazon ECS.
- Pour les tâches Amazon ECS sur EC2, les informations suivantes doivent être prises en compte :
  - Pour injecter un secret à l'aide d'une clé JSON spécifique ou d'une version d'un secret, votre instance de conteneur doit avoir la version `1.37.0` ou ultérieure de l'agent de conteneur. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

    Pour injecter le contenu complet d'un secret en tant que variable d'environnement ou pour injecter un secret dans une configuration de journal, votre instance de conteneur doit avoir la version `1.22.0` ou ultérieure de l'agent de conteneur.
- Utilisez les points de terminaison VPC de l'interface pour améliorer les contrôles de sécurité. Vous devez créer les points de terminaison de VPC d’interface pour Systems Manager. Pour plus

d'informations sur le point de terminaison de VPC, consultez la section Amélioration de la sécurité des instances EC2 à l'aide des points de terminaison de VPC pour Systems Manager dans le Guide de l'utilisateur AWS Systems Manager .

- Votre définition de tâche doit utiliser un rôle d'exécution de tâche doté des autorisations supplémentaires pour le magasin de paramètres Systems Manager. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.
- Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante :

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
Initialize-ECSAgent -Cluster <cluster name> -EnableTaskIAMRole -LoggingDrivers
 '["json-file","awslogs"]'
</powershell>
```

Création du paramètre Systems Manager

Vous pouvez utiliser la console System Manager pour créer un paramètre Systems Manager Parameter Store pour vos données sensibles. Pour plus d'informations, consultez Créer un paramètre Systems Manager (console) ou Créer un paramètre Systems Manager (AWS CLI) dans le Guide de l'utilisateur AWS Systems Manager .

Ajoutez la variable d'environnement à la définition du conteneur

Dans la définition de votre conteneur dans la définition de tâche, spécifiez `secrets` avec le nom de la variable d'environnement à définir dans le conteneur et l'ARN complet du paramètre du magasin de paramètres Systems Manager contenant les données sensibles à présenter au conteneur. Pour plus d'informations, consultez `secrets`.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store. Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
{
  "containerDefinitions": [{
```

```
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Pour plus d'informations sur la création d'une définition de tâche avec le secret spécifié dans une variable d'environnement, consultez la section Création d'une définition de tâche Amazon ECS à l'aide de la console.

Mettez à jour votre application pour récupérer par programme les secrets Systems Manager Parameter Store.

Pour récupérer les données sensibles stockées dans le paramètre Systems Manager Parameter Store, voir Exemples de code pour Systems Manager utilisant AWS SDK dans le AWS Exemples de code SDK Bibliothèque de codes.

Transmission des secrets pour la configuration de la journalisation Amazon ECS

Vous pouvez utiliser le paramètre `secretOptions` dans `logConfiguration` pour transmettre les données sensibles utilisées pour la journalisation.

Vous pouvez stocker le secret dans Secrets Manager ou Systems Manager.

Utilisation de Secrets Manager

Dans votre définition de conteneur, lorsque vous spécifiez une `logConfiguration`, vous pouvez spécifier `secretOptions` avec le nom de l'option du pilote de journal à définir dans le conteneur et l'ARN complet du secret Secrets Manager contenant les données sensibles à présenter au conteneur. Pour plus d'informations sur la création de secrets, consultez la section Création d'un AWS Secrets Manager.

Voici un extrait d'une définition de tâche montrant le format lorsque vous référencez un secret Secrets Manager.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "splunk",
      "options": {
        "splunk-url": "https://your_splunk_instance:8088"
```

```
      },
      "secretOptions": [{
        "name": "splunk-token",
        "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name-
AbCdEf"
      }]
    }]
  }]
}
```

Ajoutez la variable d'environnement à la définition du conteneur

Dans votre définition de conteneur, spécifiez `secrets` avec le nom de la variable d'environnement à définir dans le conteneur et l'ARN complet du paramètre Systems Manager Parameter Store contenant les données sensibles à présenter au conteneur. Pour plus d'informations, consultez `secrets`.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store. Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
{
  "containerDefinitions": [{
    "secrets": [{
      "name": "environment_variable_name",
      "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }]
  }]
}
```

Pour plus d'informations sur la création d'une définition de tâche avec le secret spécifié dans une variable d'environnement, consultez la section Création d'une définition de tâche Amazon ECS à l'aide de la console.

Utiliser Systems Manager

Vous pouvez injecter des données sensibles dans une configuration de journal. Dans votre définition de conteneur, lorsque vous spécifiez `logConfiguration`, vous pouvez spécifier `secretOptions` avec le nom de l'option du pilote de journal à définir dans le conteneur et l'ARN complet du paramètre Systems Manager Parameter Store contenant les données sensibles à présenter au conteneur.

> **Important**
>
> Si le paramètre Systems Manager Parameter Store existe dans la même région que la tâche que vous lancez, vous pouvez utiliser le nom ou l'ARN complet du paramètre. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

Voici un extrait d'une définition de tâche montrant le format à utiliser lorsque vous référencez un paramètre Systems Manager Parameter Store.

```
{
  "containerDefinitions": [{
    "logConfiguration": [{
      "logDriver": "fluentd",
      "options": {
        "tag": "fluentd demo"
      },
      "secretOptions": [{
        "name": "fluentd-address",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter:/parameter_name"
      }]
    }]
  }]
}
```

Spécification de données sensibles à l’aide de secrets Secrets Manager dans Amazon ECS

Amazon ECS vous permet d'injecter des données sensibles dans vos conteneurs en stockant vos données sensibles dans AWS Secrets Manager secrets, puis en les référençant dans la définition de votre conteneur. Pour plus d’informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

Découvrez comment créer un secret Secrets Manager, référencer le secret dans une définition de tâche Amazon ECS, puis vérifier que cela a fonctionné en interrogeant la variable d’environnement dans un conteneur montrant le contenu du secret.

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur dispose des autorisations IAM nécessaires pour créer les ressources Secrets Manager et Amazon ECS.

Étape 1 : Créer un secret Secrets Manager

Vous pouvez utiliser la console Secrets Manager afin de créer un secret pour vos données sensibles. Dans ce didacticiel, nous allons créer un secret de base pour stocker un nom d'utilisateur et un mot de passe à référencer ultérieurement dans un conteneur. Pour plus d'informations, voir Créez un AWS Secrets Manager secret dans le AWS Secrets Manager Guide de l'utilisateur.

Le key/value paires à stocker dans ce secret est la valeur de la variable d'environnement dans votre conteneur à la fin du didacticiel.

Enregistrez l'ARN secret pour le référencer dans votre politique IAM d'exécution de tâche et dans la définition de tâche lors des étapes ultérieures.

Étape 2 : ajouter des autorisations secrètes au rôle d’exécution de tâche

Pour qu’Amazon ECS puisse récupérer les données sensibles à partir de votre secret Secrets Manager, vous devez disposer des autorisations de secrets pour le rôle d’exécution des tâches. Pour plus d’informations, consultez Autorisations Secrets Manager ou Systems Manager.

Étape 3 : Créer une définition de tâche

Vous pouvez utiliser la console Amazon ECS pour créer une définition de tâche qui fait référence à un secret Secrets Manager.

Pour créer une définition de tâche qui spécifie un secret

Utilisez la console IAM pour mettre à jour votre rôle d'exécution de tâche avec les autorisations requises.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l’éditeur JSON, saisissez le texte JSON de définition de tâche suivant, en veillant à spécifier l’ARN complet du secret Secrets Manager que vous avez créée à l’étape 1 et le rôle d’exécution de tâche que vous avez mis à jour à l’étape 2. Cliquez sur Enregistrer.

5.

```
{
    "executionRoleArn": "arn:aws:iam::aws_account_id:role/ecsTaskExecutionRole",
    "containerDefinitions": [
        {
            "entryPoint": [
                "sh",
                "-c"
            ],
            "portMappings": [
                {
                    "hostPort": 80,
                    "protocol": "tcp",
                    "containerPort": 80
                }
            ],
            "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\""
            ],
            "cpu": 10,
            "secrets": [
                {
                    "valueFrom":
 "arn:aws:secretsmanager:region:aws_account_id:secret:username_value",
                    "name": "username_value"
                }
            ],
            "memory": 300,
            "image": "public.ecr.aws/docker/library/httpd:2.4",
            "essential": true,
            "name": "ecs-secrets-container"
        }
    ],
    "family": "ecs-secrets-tutorial"
}
```

6. Cliquez sur Créer.

Étape 4 : créer un cluster

Vous pouvez utiliser la console Amazon ECS pour créer un cluster contenant une instance de conteneur pour exécuter la tâche. Si vous avez un cluster existant avec au moins une instance de conteneur enregistrée avec les ressources disponibles pour exécuter une instance de la définition de tâche créée pour ce didacticiel, vous pouvez passer à l'étape suivante.

Pour ce didacticiel, nous allons créer un cluster avec un `t2.micro` instance de conteneur utilisant Amazon ECS-optimized AMI Amazon Linux 2.

Pour plus d'informations sur la création d'un cluster pour EC2, consultez la section the section called "Création d'un cluster pour les charges de travail Amazon EC2".

Étape 5 : exécuter une tâche

Vous pouvez utiliser la console Amazon ECS pour exécuter une tâche avec la définition de tâche que vous avez créée. Dans le cadre de ce didacticiel, nous allons exécuter une tâche utilisant EC2, à l'aide du cluster que nous avons créé lors de l'étape précédente.

Pour plus d'informations sur l'exécution d'une tâche, consultez the section called "Exécution d'une application en tant que tâche".

Étape 6 : Vérification

Vous pouvez vérifier que toutes les étapes ont été effectuées avec succès et que la variable d'environnement a été créée correctement dans votre conteneur en suivant les étapes ci-dessous.

Vérifier que la variable d'environnement a été créée

1. Trouvez l'adresse IP publique ou DNS pour votre instance de conteneur.
    a. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
    b. Dans le volet de navigation, choisissez Clusters, puis choisissez le cluster que vous avez créé.
    c. Choisissez Infrastructure, puis choisissez l'instance de conteneur.
    d. Enregistrez l'adresse Public IP (IP publique) ou Public DNS (DNS public) de votre instance.
2. Si vous utilisez un ordinateur MacOS ou Linux, connectez-vous à votre instance avec la commande suivante, en indiquant le chemin d'accès de votre clé privée et l'adresse publique de votre instance :

```
$ ssh -i /path/to/my-key-pair.pem ec2-user@ec2-198-51-100-1.compute-1.amazonaws.com
```

Pour plus d'informations sur l'utilisation d'un ordinateur Windows, consultez la section Se connecter à votre instance Linux à l'aide de PuTTY dans le Guide de l'utilisateur Amazon EC2.

> **Important**
>
> Pour plus d'informations sur les problèmes que vous pouvez rencontrer lors de la connexion à votre instance, consultez la section Résolution des problèmes de connexion à votre instance dans le Guide de l'utilisateur Amazon EC2.

3. Répertoriez les conteneurs en cours d'exécution sur l'instance. Notez l'ID de conteneur pour le conteneur `ecs-secrets-tutorial`.

```
docker ps
```

4. Connectez-vous au conteneur `ecs-secrets-tutorial` à l'aide de l'ID conteneur à partir du résultat de l'étape précédente.

```
docker exec -it container_ID /bin/bash
```

5. Utilisez la commande `echo` pour imprimer la valeur de la variable d'environnement.

```
echo $username_value
```

Si le didacticiel s'est correctement déroulé, vous devriez voir le résultat suivant :

```
password_value
```

> **Note**
>
> Sinon, vous pouvez répertorier toutes les variables d'environnement dans votre conteneur à l'aide de la commande `env` (ou `printenv`).

Étape 7 : nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Nettoyer les ressources.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Choisissez Delete Cluster (Supprimer le cluster).
5. Dans la case de confirmation, entrez supprimer ***`cluster name`***, puis choisissez Supprimer.
6. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
7. Dans le panneau de navigation, choisissez Rôles.
8. Recherchez `ecsTaskExecutionRole` dans la liste des rôles et sélectionnez-le.
9. Choisissez Permissions (Autorisations), puis choisissez le X en regard de ECSSecretsTutorial. Cliquez sur Supprimer.
10. Ouvrez la console Secrets Manager à l'adresse https://console.aws.amazon.com/secretsmanager/.
11. Sélectionnez le secret username_value que vous avez créé, puis choisissez Actions, Delete secret (Supprimer le secret).

Paramètres de définition de tâche Amazon ECS pour les instances gérées Amazon ECS

Les définitions de tâches sont divisées en plusieurs parties : la famille de tâches, Gestion des identités et des accès AWS le rôle des tâches (IAM), le mode réseau, les définitions des conteneurs, les volumes et la capacité. Les définitions de famille et de conteneur sont requises dans une définition de tâche. En revanche, le rôle de tâche, le mode réseau, les volumes et la capacité sont facultatifs.

Vous pouvez utiliser ces paramètres dans un fichier JSON pour configurer votre définition de tâche.

Vous trouverez ci-dessous une description plus détaillée de chaque paramètre de définition de tâche pour les instances gérées Amazon ECS.

Family

`family`

Type : chaîne

Obligatoire : oui

Lorsque vous enregistrez une définition de tâche, vous lui attribuez une famille. Cela équivaut, pour plusieurs versions de définition de tâche, à lui attribuer un nom spécifié avec un numéro de révision. La première définition de tâche enregistrée dans une famille donnée reçoit le numéro de révision 1. Toute définition de tâche enregistrée après celle-ci reçoit un numéro de révision ultérieur dans l'ordre séquentiel.

Capacity

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la capacité par rapport à laquelle Amazon ECS doit valider la définition de tâche. Une exception client est renvoyée si la définition de tâche n'est pas conforme aux compatibilités spécifiées. Pour plus d'informations, consultez la section Types de lancement Amazon ECS.

Le paramètre suivant est autorisé dans une définition de tâche.

`requiresCompatibilities`

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `MANAGED_INSTANCES`

La capacité par rapport à laquelle valider la définition de tâche. Une vérification est alors lancée afin de s'assurer que tous les paramètres utilisés dans la définition de tâche répondent aux exigences des instances gérées Amazon ECS.

Rôle de tâche

`taskRoleArn`

Type : chaîne

Obligatoire : non

Lorsque vous enregistrez une définition de tâche, vous pouvez fournir un rôle de tâche pour un rôle IAM qui permet aux conteneurs dans l'autorisation de tâche d'appeler en votre nom les API AWS qui sont spécifiées dans les stratégies qui lui associées. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Rôle d'exécution de tâche

`executionRoleArn`

Type : String

Obligatoire : Conditionnelle

Le nom de ressource Amazon (ARN) du rôle d'exécution de la tâche qui autorise l'agent de conteneur Amazon ECS à effectuer AWS Appels d'API en votre nom. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Note**
>
> Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Le rôle est requis pour les extractions d'images ECR privées et l'utilisation du `awslogs` pilote de journaux.

Mode réseau

`networkMode`

Type : chaîne

Obligatoire : non

Valeur par défaut : `awsvpc`

Mode réseau à utiliser pour les conteneurs de la tâche. Pour les tâches Amazon ECS hébergées sur des instances gérées Amazon ECS, les valeurs valides sont `awsvpc` et `host`. Si aucun mode réseau n'est spécifié, le mode réseau par défaut est `awsvpc`.

Si le mode réseau est `host`, la tâche contourne l'isolation du réseau et les conteneurs utilisent directement la pile réseau de l'hôte.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, vous n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0) pour une meilleure sécurité. Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Si le mode réseau est `awsvpc`, une interface réseau Elastic est attribuée à la tâche, et vous devez spécifier une configuration `NetworkConfiguration` lorsque vous créez un service ou exécutez une tâche avec la définition de tâche. Pour plus d'informations, consultez Mise en réseau des tâches Amazon ECS pour les instances gérées Amazon ECS.

Les modes réseau `host` et `awsvpc` offrent les meilleures performances de mise en réseau pour les conteneurs, car ils utilisent la pile de réseau Amazon EC2. Avec les modes réseau `host` et `awsvpc`, les ports de conteneur exposés sont mappés directement au port hôte correspondant (pour le mode réseau hôte `host`) ou au port de l'interface réseau Elastic (pour le mode réseau `awsvpc`). Pour cette raison, vous ne pouvez pas utiliser de mappages de ports hôtes dynamiques.

Plateforme d'exécution

`operatingSystemFamily`

Type : chaîne

Obligatoire : non

Par défaut : LINUX

Lorsque vous enregistrez une définition de tâche, vous spécifiez la famille du système d'exploitation.

La valeur valide pour ce champ est `LINUX`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsqu'une définition de tâche fait partie d'un service, cette valeur doit correspondre à la valeur `platformFamily` du service.

`cpuArchitecture`

Type : String

Obligatoire : Conditionnelle

Lorsque vous enregistrez une définition de tâche, vous spécifiez l'architecture du processeur. Les valeurs valides sont `X86_64` et `ARM64`.

Si vous ne spécifiez aucune valeur, Amazon ECS tente de placer les tâches sur l'architecture de processeur disponible en fonction de la configuration du fournisseur de capacité. Pour vous assurer que les tâches sont placées sur une architecture de processeur spécifique, spécifiez une valeur pour `cpuArchitecture` dans la définition de la tâche.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Pour plus d'informations sur `ARM64`, consultez the section called “Définitions de tâche pour les charges de travail ARM 64 bits”.

Taille de la tâche

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la quantité totale d'UC et de mémoire utilisée pour cette tâche. Ces valeurs sont distinctes des valeurs `cpu` et `memory` au niveau de la définition de conteneur. Pour les tâches hébergées sur des instances Amazon EC2, ces champs sont facultatifs.

> **Note**
>
> Task-level Les paramètres du processeur et de la mémoire sont ignorés pour les conteneurs Windows. Nous vous recommandons de spécifier des ressources de niveau conteneur pour les conteneurs Windows.

`cpu`

Type : String

Obligatoire : Conditionnelle

Limite stricte du nombre d'unités UC à présenter pour la tâche. Vous pouvez spécifier les valeurs de l'UC dans le fichier JSON sous forme de chaîne en unités d'UC ou en processeurs virtuels (vCPU). Par exemple, vous pouvez spécifier une valeur d'UC de `1024` en unités d'UC ou de `1 vCPU` en vCPU. Lorsque la définition de tâche est enregistrée, la valeur de vCPU (processeurs virtuels) est convertie en un nombre entier indiquant les unités UC.

Ce champ est facultatif. Si votre cluster n'a pas d'instances de conteneur enregistrées avec les unités UC demandées disponibles, la tâche échoue. Les valeurs prises en charge se situent entre `0.125` vCPU et `10` vCPU.

`memory`

Type : String

Obligatoire : Conditionnelle

Limite stricte de quantité de mémoire à allouer à la tâche. Vous pouvez spécifier les valeurs de mémoire dans la définition de la tâche sous forme de chaîne en mébioctets (Mio) ou en gigaoctets (Go). Par exemple, vous pouvez spécifier une valeur de mémoire de `3072` en Mio ou de `3 GB` en Go. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Ce champ est facultatif et n'importe quelle valeur peut être utilisée. Si une valeur de mémoire au niveau de la tâche est spécifiée, la valeur de mémoire au niveau du conteneur est facultative. Si votre cluster n'a pas d'instances de conteneur enregistrées avec la mémoire demandée disponible, la tâche échoue. Vous pouvez optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier. Pour plus d'informations, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

Autres paramètres de définition de tâche

Les paramètres de définition de tâche suivants peuvent être utilisés lors de l'enregistrement des définitions de tâches dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Stockage éphémère
- Mode IPC
- Mode PID
- Configuration du proxy
- Étiquettes
- Accélérateur d'inférence élastique (obsolète)
- Contraintes de placement
- Volumes

Stockage éphémère

`ephemeralStorage`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : objet EphemeralStorage

Obligatoire : non

Quantité de stockage éphémère (en Go) à allouer pour la tâche. Ce paramètre est utilisé pour étendre la quantité totale de stockage éphémère disponible, au-delà de la quantité par défaut, pour les tâches hébergées sur AWS Fargate. Pour plus d'informations, consultez the section called "Montages liés".

Mode IPC

`ipcMode`

Type : chaîne

Obligatoire : non

Espace de noms de ressource IPC à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host`, `task` ou `none`. Si `host` est spécifié, tous les conteneurs des tâches ayant spécifié le mode IPC `host` sur la même instance de conteneur partagent les mêmes ressources IPC avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent les mêmes ressources IPC. Si `none` est spécifié, les ressources IPC des conteneurs d'une tâche sont privés et ne partagent rien avec les autres conteneurs d'une tâche ou d'une instance de conteneur. Si aucune valeur n'est spécifiée, le partage de l'espace de noms des ressources IPC dépend de la configuration d'exécution du conteneur.

Mode PID

`pidMode`

Type : chaîne

Obligatoire : non

Espace de noms de processus à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host` ou `task`. Si `host` est spécifié, tous les conteneurs qui se trouvent dans les tâches ayant spécifié le mode PID `host` sur la même instance de conteneur partagent le même espace de noms de processus avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs qui se trouvent dans la tâche spécifiée partagent le même espace de noms de processus. Si aucune valeur n'est spécifiée, la valeur par défaut est un espace de noms privé.

Si le mode PID `host` est utilisé, il existe un risque accru d'exposition à un espace de noms de processus indésirable.

Configuration du proxy

`proxyConfiguration`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : objet ProxyConfiguration

Obligatoire : non

Détails de configuration pour le proxy App Mesh.

Étiquettes

Métadonnées que vous appliquez à une définition de tâche pour faciliter le classement et l'organisation. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous définissez ces deux éléments.

Les restrictions de base suivantes s'appliquent aux balises :

- Nombre maximal de balises par ressource - 50
- Pour chaque ressource, chaque clé d'identification doit être unique, et chaque clé d'identification peut avoir une seule valeur.
- Longueur de clé maximale : 128 caractères Unicode UTF-8
- Longueur de valeur maximale : 256 caractères Unicode UTF-8
- Si votre schéma de balisage est utilisé pour plusieurs services et ressources , n'oubliez pas que d'autres services peuvent avoir des restrictions concernant les caractères autorisés. Les caractères généralement autorisés sont les suivants : lettres, chiffres et espaces représentables dans UTF-8, et les caractères suivants : + - =. _ :/@.
- Les clés et valeurs d'étiquette sont sensibles à la casse.
- Ne pas utiliser `aws:`, `AWS:`, ou toute combinaison de majuscules ou de minuscules, telle qu'un préfixe pour les clés ou les valeurs, car il est réservé AWS utiliser. Vous ne pouvez pas modifier ni supprimer des clés ou valeurs d'étiquette ayant ce préfixe. Les étiquettes avec ce préfixe ne sont pas comptabilisées comme vos étiquettes pour la limite de ressources.

`key`

Type : chaîne

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : chaîne

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

Accélérateur d'inférence élastique (obsolète)

> **Note**
>
> Amazon Elastic Inference (EI) n’est plus disponible pour les clients.

`inferenceAccelerator`

> **Note**
>
> Ce paramètre n’est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : objet InferenceAccelerator

Obligatoire : non

Accélérateurs d'inférence élastique à utiliser pour les conteneurs de la tâche.

Contraintes de placement

`placementConstraints`

Type : tableau d’objets TaskDefinitionPlacementConstraint

Obligatoire : non

Un tableau d'objets de contraintes de placement à utiliser pour la tâche. Vous pouvez spécifier 10 contraintes maximum par tâche (cette limite comprend les contraintes de la définition de tâche et celles spécifiées lors de l'exécution).

Amazon ECS prend en charge les contraintes de placement `distinctInstace` et `memberOf` pour les tâches exécutées sur les instances gérées Amazon ECS. Les attributs suivants sont pris en charge pour les tâches qui utilisent la contrainte de placement `memberOf` :

- `ecs.subnet-id`
- `ecs.availability-zone`
- `ecs.cpu-architecture`
- `ecs.instance-type`

Pour plus d'informations sur les contraintes de placement, consultez la section Définition des instances de conteneur utilisées par Amazon ECS pour les tâches.

Volumes

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier une liste de volumes pour vos tâches. Cela vous permet d'utiliser des volumes de données dans vos tâches.

Pour de plus amples informations sur les types de volume et d'autres paramètres, consultez la section the section called "Options de stockage pour les tâches".

`name`

Type : chaîne

Obligatoire : oui

Le nom du volume. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres et des tirets. Ce nom est référencé dans le `sourceVolume` paramètre de définition de conteneur `mountPoints`.

`host`

Type : objet HostVolumeProperties

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes hôtes de montage lié. Le contenu du paramètre `host` détermine si votre volume hôte de montage lié persiste sur l'instance de conteneur hôte et où il est stocké. Si le `host` Si le paramètre est vide, le système attribue

un chemin d'hôte à votre volume de données. Toutefois, il n'est pas garanti que les données perdureront après l'arrêt de l'exécution des conteneurs associés.

`sourcePath`

Type : chaîne

Obligatoire : non

Quand le `host` le paramètre est utilisé, spécifiez un `sourcePath` pour déclarer le chemin sur l'instance hôte qui est présentée au conteneur. Si ce paramètre est vide, le système vous attribue un chemin d'hôte. Si le `host` le paramètre contient un `sourcePath` emplacement du fichier, puis le volume de données persiste à l'emplacement spécifié sur l'instance hôte jusqu'à ce que vous le supprimiez manuellement. Si le `sourcePath` la valeur n'existe pas sur l'instance hôte, le système la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

Sur les instances gérées Amazon ECS, certaines parties du système de fichiers hôte sont en lecture seule. Le `sourcePath` doit pointer vers un répertoire accessible en écriture tel que `/var` ou `/tmp`. Pour plus d'informations, consultez Utilisation de montages liés avec Amazon ECS.

`dockerVolumeConfiguration`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : objet DockerVolumeConfiguration

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker.

`efsVolumeConfiguration`

Type : objet EFSVolumeConfiguration

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon EFS pour le stockage des tâches.

`fsxWindowsFileServerVolumeConfiguration`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : objet FSxWindowsFileServerVolumeConfiguration

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez le système de fichiers Amazon FSx for Windows File Server pour le stockage des tâches.

`configuredAtLaunch`

Type : booléen

Obligatoire : non

Indique si le volume doit être configuré lors du lancement. Cette méthode est utilisée pour créer des volumes Amazon EBS pour des tâches autonomes ou des tâches créées dans le cadre d'un service. Chaque révision de définition de tâche ne peut comporter qu'un seul volume configuré lors du lancement dans la configuration du volume.

Définitions de conteneur

Lorsque vous enregistrez une définition de tâche, vous devez spécifier une liste de définitions de conteneur qui sont transmises au démon Docker sur une instance de conteneur. Les paramètres suivants sont autorisés dans une définition de conteneur.

Rubriques

- Nom
- Image
- Mémoire

- CPU
- Cartographie des ports
- Informations d'identification du référentiel privé
- Essential
- Point d'entrée
- Commande
- Répertoire de travail
- Paramètres de définition de conteneur avancés
- Paramètres Linux

Nom

`name`

Type : chaîne

Obligatoire : oui

Le nom d'un conteneur. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés. Si vous associez plusieurs conteneurs dans une définition de tâche, vous pouvez spécifier l'option `name` d'un conteneur dans l'option `links` d'un autre conteneur. Cela permet de connecter les conteneurs.

Image

`image`

Type : chaîne

Obligatoire : oui

Image utilisée pour démarrer un conteneur. Cette chaîne est transmise directement au démon Docker. Par défaut, les images dans le registre Docker Hub sont disponibles. Vous pouvez également spécifier d'autres référentiels avec soit *`repository-url`*`/`*`image`*`:`*`tag`*, soit *`repository-url`*`/`*`image`*`@`*`digest`*. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, deux points, des points, des

barres obliques et des signes dièse. Ce paramètre correspond à `Image` dans la commande docker create-container et `IMAGE` paramètre de la commande Docker run.

- Lorsqu'une nouvelle tâche démarre, l'agent de conteneur Amazon ECS extrait la version la plus récente de l'image et de l'étiquette spécifiées afin que le conteneur puisse les utiliser. Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Lorsque vous ne spécifiez pas de balise ou de résumé dans le chemin de l'image dans la définition de la tâche, l'agent de conteneur Amazon ECS utilise le `latest` balise pour extraire l'image spécifiée.
- Les mises à jour ultérieures apportées à une image de référentiel ne sont pas propagées aux tâches déjà en cours d'exécution.
- Les images des registres privés sont prises en charge. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.
- Les images des référentiels Amazon ECR peuvent être spécifiées en utilisant soit la convention de dénomination complète `registry/repository:tag` ou `registry/repository@digest` (par exemple, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Type : String

Valeurs valides : `enabled|disabled`

Obligatoire : non

Spécifie si Amazon ECS résoudra la balise d'image de conteneur fournie dans la définition du conteneur en un condensé d'image. Par défaut, ce comportement est `enabled`. Si vous définissez la valeur d'un conteneur sur `disabled`, Amazon ECS ne résoudra pas la balise

d'image de conteneur en un condensé et utilisera l'URI d'image d'origine spécifiée dans la définition du conteneur pour le déploiement. Pour plus d'informations sur la résolution des images de conteneur, consultez la section Résolution des images de conteneurs.

Mémoire

`memory`

Type : Integer

Obligatoire : non

La quantité de mémoire (en Mio) à présenter au conteneur. Si votre conteneur tente de dépasser la mémoire spécifiée ici, il sera désactivé. La quantité totale de mémoire réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `memory` de la tâche, si cette valeur est spécifiée. Ce paramètre correspond à `Memory` dans la commande docker create-container et à l'option `--memory` dans la commande docker run.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

`memoryReservation`

Type : Integer

Obligatoire : non

La limite flexible (en Mio) de mémoire à réserver pour le conteneur. En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Toutefois, votre conteneur peut utiliser davantage de mémoire en cas de besoin. Le

conteneur peut consommer jusqu'à la limite stricte spécifiée avec le paramètre `memory` (le cas échéant), ou la totalité de la mémoire disponible sur l'instance de conteneur, selon la première valeur atteinte. Ce paramètre correspond à `MemoryReservation` dans la commande docker create-container et à l'option `--memory-reservation` dans la commande docker run.

Si aucune valeur de mémoire au niveau de la tâche n'est spécifiée, vous devez indiquer un nombre entier différent de zéro pour `memory` ou `memoryReservation` dans une définition de conteneur. Si vous spécifiez les deux, `memory` doit être supérieur à `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Par exemple, supposons que votre conteneur utilise normalement 128 Mio de mémoire, mais qu'il lui arrive d'utiliser jusqu'à 256 Mio de mémoire pendant de courtes périodes. Vous pouvez définir une `memoryReservation` de 128 Mio et une limite stricte `memory` de 300 Mio. Cette configuration permet au conteneur de ne réserver que 128 Mio de mémoire à partir des ressources restantes sur l'instance de conteneur. En même temps, cette configuration permet également au conteneur d'utiliser davantage de ressources mémoire lorsque cela est nécessaire.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

CPU

`cpu`

Type : Integer

Obligatoire : non

Le nombre d'unités `cpu` réservées pour le conteneur. Ce paramètre correspond à `CpuShares` dans la commande docker create-container et à l'option `--cpu-shares` dans la commande docker run.

Ce champ est facultatif pour les tâches utilisant des fournisseurs de capacité EC2. La seule exigence est que la quantité totale d'UC réservée pour tous les conteneurs d'une tâche soit inférieure à la valeur `cpu` au niveau de la tâche.

> **Note**
>
> Vous pouvez déterminer le nombre d'unités UC qui sont disponibles par type d'instance EC2 en multipliant par 1024 le nombre de processeurs virtuels (vCPU) répertoriés pour ce type d'instance sur la page des détails des instances Amazon EC2.

Les conteneurs Linux partagent des unités d'UC non allouées avec les autres conteneurs de l'instance de contenu selon le même ratio que la quantité allouée. Par exemple, si vous exécutez une tâche à un seul conteneur sur un type d'instance à un seul cœur avec 512 unités d'UC spécifiées pour ce conteneur, et que c'est la seule tâche en cours d'exécution sur l'instance de conteneur, ce conteneur pourrait utiliser la totalité de la totalité des 1 024 unités d'UC à tout moment. Toutefois, si vous avez lancé une autre copie de la même tâche sur cette instance de conteneur, chaque tâche se voit garantir un minimum de 512 unités de processeur si nécessaire. De plus, chaque conteneur pourrait bénéficier d'une utilisation plus importante du processeur si l'autre conteneur ne l'utilisait pas. Toutefois, si les deux tâches étaient actives à 100 % en permanence, elles seraient limitées à 512 unités de processeur.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur d'UC pour calculer les ratios de partage d'UC relatifs pour les conteneurs en cours d'exécution. Pour plus d'informations, consultez la section CPU share constraint dans la documentation Docker. La valeur de parts d'UC minimale autorisée par le noyau Linux est de 2. Toutefois, le paramètre de processeur n'est pas obligatoire et vous pouvez utiliser des valeurs de processeur inférieures à 2 dans vos définitions de conteneur. Pour les valeurs d'UC inférieures à 2 (y compris null), le comportement varie selon la version de l'agent de conteneur Amazon ECS :

- Versions d'agent inférieures ou égales à 1.1.0 : Les valeurs d'UC null et égales à zéro sont transmises à Docker en tant que 0, ce que Docker convertit ensuite en 1024 parts d'UC. Les valeurs d'UC de 1 sont transmises à Docker en tant que 1, ce que le noyau Linux convertit en deux parts d'UC.

- Versions d'agent supérieures ou égales à 1.2.0 : Les valeurs d'UC null, et égales à zéro et à 1 sont transmises à Docker en tant que 2.

Sur les instances de conteneur Windows, la limite d'UC est appliquée comme limite absolue ou quota. Les conteneurs Windows n'ont accès qu'à la quantité spécifiée de processeur décrite dans la définition de tâche. Une valeur d'UC null ou nulle est transmise à Docker comme `0`, que Windows interprète comme 1 % d'un processeur.

Cartographie des ports

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports exposent les ports réseau de votre conteneur au monde extérieur, ce qui permet aux clients d'accéder à votre application. Ils sont également utilisés pour la communication entre conteneurs au sein d'une même tâche.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le `hostPort` est toujours ignoré et le port du conteneur est automatiquement mappé à un port aléatoire portant un numéro élevé sur l'hôte.

La plupart des champs de ce paramètre (y compris `containerPort`, `hostPort`, `protocol`) carte vers `PortBindings` dans la commande docker create-container et `--publish` option pour exécuter Docker. Si le mode réseau d'une définition de tâche est défini sur `host`, les ports hôtes doivent être indéfinis ou correspondre au port du conteneur dans le mappage de port.

> **Note**
>
> Une fois qu'une tâche passe à l'état `RUNNING`, les affectations manuelles et automatiques de ports de conteneur et d'hôte sont visibles aux emplacements suivants :
>
> - Console : la section Network Bindings (Liaisons réseau) de la description d'un conteneur pour une tâche sélectionnée.
> - AWS CLI : la section `networkBindings` de la sortie de la commande describe-tasks.
> - API : la réponse `DescribeTasks`.
> - Métadonnées : point de terminaison des métadonnées de la tâche.

`appProtocol`

Type : chaîne

Obligatoire : non

Protocole d'application utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Nous vous conseillons de définir ce paramètre de manière cohérente avec le protocole que votre application utilise. Si vous définissez ce paramètre, Amazon ECS ajoute une gestion de connexion spécifique au protocole au proxy Service Connect. Si vous définissez ce paramètre, Amazon ECS ajoute une télémétrie spécifique au protocole dans la console Amazon ECS et CloudWatch.

Si vous ne définissez aucune valeur pour ce paramètre, le protocole TCP est utilisé. Toutefois, Amazon ECS n'ajoute pas de télémétrie spécifique au protocole pour TCP.

Pour plus d'informations, consultez the section called “ Connexion au service”.

Valeurs de protocole valides : `"HTTP" | "HTTP2" | "GRPC"`

`containerPort`

Type : Integer

Obligatoire : oui, lorsque des objets `portMappings` sont utilisés

Le numéro de port sur le conteneur qui est lié au port hôte spécifié par l'utilisateur ou affecté automatiquement.

Pour les tâches utilisant le mode réseau `awsvpc`, vous utilisez `containerPort` pour spécifier les ports exposés.

`containerPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement.

Vous ne pouvez définir ce paramètre qu'à l'aide de l'API `register-task-definition`. L'option est disponible dans le paramètre `portMappings`. Pour plus d'informations, consultez register-task-definition dans la Référence AWS Command Line Interface .

Les règles suivantes s'appliquent lorsque vous spécifiez une `containerPortRange` :

- Vous devez utiliser le mode réseau `awsvpc`.
- L'instance de conteneur doit au moins disposer de la version 1.67.0 de l'agent de conteneur et au moins de la version 1.67.0-1 du package `ecs-init`.
- Vous pouvez spécifier un maximum de 100 plages de ports pour chaque conteneur.
- Vous ne spécifiez pas de `hostPortRange`. La valeur de `hostPortRange` est définie comme suit :
  - Pour les conteneurs d'une tâche en mode réseau `awsvpc`, le `hostPort` est défini sur la même valeur que le `containerPort`. Il s'agit d'une stratégie de mappage statique.
- Les valeurs valides de `containerPortRange` sont comprises entre 1 et 65535.
- Un port peut uniquement être inclus dans un mappage de port pour chaque conteneur.
- Vous ne pouvez pas spécifier de plages de ports qui se chevauchent.
- Le premier port de la plage doit être inférieur au dernier port de la plage.
- Docker vous recommande de désactiver le Docker-proxy dans le Docker fichier de configuration du démon lorsque vous avez un grand nombre de ports.

  Pour plus d'informations, voir Numéro #11185 sur GitHub.

  Pour plus d'informations sur la désactivation du proxy Docker dans le fichier de configuration du démon Docker, veuillez consulter Démon Docker dans le Guide du développeur Amazon ECS (langue française non garantie).

Vous pouvez appeler DescribeTasks pour voir la `hostPortRange` correspondant aux ports hôtes liés aux ports de conteneur.

Les plages de ports ne sont pas incluses dans les événements de tâches Amazon ECS, qui sont envoyés à EventBridge. Pour plus d'informations, consultez the section called "Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge".

`hostPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port sur l'hôte utilisée avec la liaison réseau. Elle est attribuée par Docker et délivrée par l'agent Amazon ECS.

`hostPort`

Type : Integer

Obligatoire : non

Le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Le `hostPort` peut rester vide ou avoir la même valeur que `containerPort`.

La plage de ports éphémères par défaut pour Docker 1.6.0 et versions ultérieures est répertoriée dans l'instance sous `/proc/sys/net/ipv4/ip_local_port_range`. Si ce paramètre de noyau n'est pas disponible, la plage de ports éphémères par défaut de `49153-65535` est utilisée. N'essayez pas de spécifier un port d'hôte dans la plage de ports éphémères. Cela est dû au fait qu'ils sont réservés à une affectation automatique. En général, les ports inférieurs à `32768` ne sont pas compris dans la plage de ports éphémères.

Les ports réservés par défaut sont le port `22` pour SSH, les ports Docker `2375` et `2376`, et les ports d'agent de conteneur Amazon ECS `51678-51680`. Tout port hôte ayant été spécifié précédemment par l'utilisateur pour une tâche en cours d'exécution est également réservé pendant l'exécution de la tâche. Après l'arrêt d'une tâche, le port hôte est libéré. Les ports réservés actuels s'affichent dans le champ `remainingResources` de la sortie describe-container-instances. Une instance de conteneur peut avoir jusqu'à 100 ports réservés à la fois, y compris les ports réservés par défaut. Les ports affectés automatiquement ne sont pas pris en compte dans le quota des 100 ports réservés.

`name`

Type : String

Obligatoire : Non, requis pour que Service Connect et VPC Lattice soient configurés dans un service

Nom utilisé pour le mappage de port. Ce paramètre ne s’applique qu’à Service Connect et VPC Lattice. Ce paramètre correspond au nom que vous utilisez dans la configuration Service Connect et VPC Lattice d’un service.

Pour plus d’informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.

Dans l’exemple suivant, les deux champs obligatoires pour Service Connect et VPC Lattice sont utilisés.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Type : chaîne

Obligatoire : non

Le protocole utilisé pour le mappage de port. Les valeurs valides sont `tcp` et `udp`. La valeur par défaut est `tcp`.

> **Important**
>
> Seul `tcp` est pris en charge pour Service Connect. N'oubliez pas que `tcp` est implicite si ce champ n'est pas défini.

Si vous spécifiez un port hôte, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Si vous souhaitez utiliser un port hôte affecté automatiquement, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
```

```
]
```

Informations d'identification du référentiel privé

`repositoryCredentials`

Type : objet RepositoryCredentials

Obligatoire : non

Informations d'identification du référentiel pour l'authentification de registre privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

`credentialsParameter`

Type : String

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS, AWS CLI, ou AWS SDK, si le secret existe dans la même région que la tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Lorsque vous utilisez Console de gestion AWS, vous devez spécifier l'ARN complet du secret.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
```

```
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Essential

`essential`

Type : booléen

Obligatoire : non

Si le paramètre `essential` d'un conteneur a la valeur `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur est marqué comme `false`, son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Si vous avez une application composée de plusieurs conteneurs, regroupez les conteneurs utilisés dans un même but et séparer les différents composants en plusieurs définitions de tâche. Pour plus d'informations, consultez the section called “Architecture de votre application.”.

Point d'entrée

`entryPoint`

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur. Ce paramètre correspond à `Entrypoint` dans la commande docker create-container et à l'option `--entrypoint` dans la commande docker run.

```
"entryPoint": ["string", ...]
```

Commande

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond à `Cmd` dans la commande docker create-container et au paramètre `COMMAND` dans docker run. S'il existe plusieurs arguments, chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

Répertoire de travail

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la commande docker create-container et à l'option `--workdir` dans la commande docker run.

Paramètres de définition de conteneur avancés

Les paramètres avancés de définition de conteneur suivants fournissent des fonctionnalités étendues à la commande Docker run qui est utilisée pour lancer des conteneurs sur vos instances de conteneurs Amazon ECS.

Rubriques

- Politique de redémarrage
- Surveillance de l'état
- Environnement
- Sécurité
- Paramètres réseau

- Stockage et journalisation
- Besoins en ressources
- Temporisations de conteneurs
- Dépendances du conteneur
- Contrôles système
- Interactive
- Pseudo Terminal

Politique de redémarrage

`restartPolicy`

La politique de redémarrage du conteneur et les paramètres de configuration associés. Lorsque vous configurez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur sans avoir à remplacer la tâche. Pour plus d'informations, consultez Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`enabled`

Type : Boolean

Obligatoire : oui

Spécifie si une politique de redémarrage est activée pour le conteneur.

`ignoredExitCodes`

Type : tableau integer

Obligatoire : non

Liste des codes de sortie qu'Amazon ECS ignorera et pour lesquels il ne tentera pas de redémarrage. Vous pouvez spécifier un maximum de 50 codes de sortie de conteneur. Par défaut, Amazon ECS n'ignore aucun code de sortie.

`restartAttemptPeriod`

Type : Integer

Obligatoire : non

Durée (en secondes) pendant laquelle le conteneur doit fonctionner avant de pouvoir tenter un redémarrage. Un conteneur ne peut être redémarré qu'une fois toutes les `restartAttemptPeriod` secondes. Si un conteneur n'est pas en mesure de fonctionner pendant cette période et qu'il se ferme prématurément, il ne sera pas redémarré.
Vous pouvez définir une `restartAttemptPeriod` minimale de 60 secondes et une `restartAttemptPeriod` maximale de 1 800 secondes. Par défaut, un conteneur doit fonctionner pendant 300 secondes avant de pouvoir être redémarré.

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour plus d'informations, consultez Détermination de l'état des tâches Amazon ECS à l'aide de la surveillance de l'état des conteneurs.

`command`

Tableau de chaînes représentant la commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé.

Lors de l'enregistrement d'une définition de tâche dans le Console de gestion AWS, utilisez une liste de commandes séparées par des virgules. Ces commandes sont converties en une chaîne une fois la définition de tâche créée. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Lors de l'enregistrement d'une définition de tâche à l'aide du Console de gestion AWS panneau JSON, le AWS CLI, ou les API, placez la liste des commandes entre crochets. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un code de sortie de 0, avec aucune sortie `stderr`, indique la réussite et un code de sortie autre que zéro indique qu'il s'agit d'un échec.

`interval`

La durée (en secondes) entre chaque surveillance de l'état. Vous pouvez indiquer une durée comprise entre 5 et 300 secondes. La valeur par défaut est de 30 secondes.

`timeout`

La durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant qu'elle ne soit considérée comme un échec. Vous pouvez indiquer une durée comprise entre 2 et 60 secondes. La valeur par défaut est de 5 secondes.

`retries`

Nombre de nouvelles tentatives d'exécution d'une surveillance de l'état ayant échoué avant que le conteneur soit considéré comme défectueux. Vous pouvez indiquer un nombre de tentatives compris en 1 et 10. La valeur par défaut est de trois tentatives.

`startPeriod`

La période de grâce facultative pour donner aux conteneurs le temps de démarrer avant l'échec des surveillance de l'état est prise en compte dans le nombre maximal de tentatives. Vous pouvez spécifier une valeur comprise entre 0 et 300. Par défaut, la `startPeriod` est désactivée.

Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Environnement

`cpu`

Type : Integer

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sous Linux, ce paramètre correspond à `CpuShares` dans la section Créer un conteneur.

Ce champ est facultatif pour les tâches exécutées sur les instances gérées Amazon ECS. La quantité totale de processeurs réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `cpu` au niveau de la tâche.

Les conteneurs Linux partagent des unités de processeur non allouées avec les autres conteneurs de l'instance de conteneur selon le même ratio que la quantité allouée. Par exemple, supposons que vous exécutez une tâche à conteneur unique sur un type d'instance à cœur unique avec 512 unités de processeur spécifiées pour ce conteneur. De plus, cette tâche est la seule tâche en cours d'exécution sur l'instance de conteneur. Dans cet exemple, le conteneur peut utiliser le partage complet de 1 024 unités UC à tout moment. Toutefois, supposons que vous ayez lancé une autre copie de la même tâche sur cette instance de conteneur. Un minimum de 512 unités de processeur est affecté à chaque tâche si nécessaire. De même, si l'autre conteneur n'utilise pas le processeur restant, chaque conteneur peut bénéficier d'une utilisation plus importante du processeur. Toutefois, si les deux tâches sont actives à 100 % en permanence, elles sont limitées à 512 unités d'UC.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur de processeur pour calculer les ratios de partage de processeur relatifs pour les conteneurs en cours d'exécution. La valeur de parts d'UC minimale autorisée par le noyau Linux est de 2, et la valeur de parts d'UC maximale autorisée par le noyau Linux est de 262 144. Toutefois, le paramètre UC n'est pas obligatoire et vous pouvez utiliser des valeurs d'UC inférieures à 2 ou supérieures à 262 144 dans vos définitions de conteneur. Pour les valeurs d'UC inférieures à 2 (y compris null) ou supérieures à 262 144, le comportement varie selon la version de l'agent de conteneur Amazon ECS :

Pour voir d'autres exemples, veuillez consulter le billet de blog How Amazon ECS manages CPU and memory resources.

gpu

Type : objet ResourceRequirement

Obligatoire : non

Le nombre de `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Vous pouvez spécifier une valeur entière, une valeur décimale pour les GPU fractionnaires ou `ALL`. Lorsque vous spécifiez `ALL`, tous les GPU de l'instance de conteneur sont alloués au conteneur. Lorsque vous spécifiez une valeur décimale (telle que `0.125`, `0.25`, ou `0.5`), Amazon ECS place la tâche sur une instance GPU fractionnée (G6f) qui fournit la capacité GPU requise. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne doit pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.

`Neuron device`

Type : objet ResourceRequirement

Obligatoire : non

Les appareils Neuron que l'agent de conteneur Amazon ECS réserve pour le conteneur. La seule valeur prise en charge est `ALL`, qui donne au conteneur un accès exclusif à tous les appareils Neuron de l'instance. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones.

`Elastic Inference accelerator`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs hébergés sur instances gérées Amazon ECS.

Type : objet ResourceRequirement

Obligatoire : non

Pour le type `InferenceAccelerator`, `value` correspond à `deviceName` d'un `InferenceAccelerator` spécifié dans une définition de tâche. Pour plus d'informations, consultez the section called "Nom de l'accélérateur Elastic Inference (obsolète)".

`essential`

Type : booléen

Obligatoire : non

Supposons que le paramètre `essential` d'un conteneur soit marqué comme `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque. Ensuite, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur a la valeur `false`, son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Supposons que vous disposiez d'une application composée de plusieurs conteneurs. Ensuite, les conteneurs de

groupes qui sont utilisés dans un même but pour former des composants, et séparer les différents composants en plusieurs définitions de tâches. Pour plus d'informations, consultez Architecture de votre application pour Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Les premières versions de l'agent de conteneur Amazon ECS ne traitent pas correctement les paramètres `entryPoint`. Si vous rencontrez des difficultés pour utiliser `entryPoint`, mettez à jour l'agent de conteneur ou saisissez plutôt vos commandes et vos arguments sous forme d'éléments de tableau `command`.

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond à `Cmd` dans la commande create-container et au paramètre `COMMAND` de docker run. S'il existe plusieurs arguments, assurez-vous que chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--workdir` correspond à docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Type : tableau d'objets

Obligatoire : non

Liste des fichiers contenant les variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `--env-file` option de la commande Docker run.

Vous pouvez spécifier jusqu'à dix fichiers d'environnement. Le fichier doit avoir une extension de fichier `.env`. Chaque ligne d'un fichier d'environnement contient une variable d'environnement au format `VARIABLE=VALUE`. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées.

Si des variables d'environnement individuelles sont spécifiées dans la définition du conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement. Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités de haut en bas. Nous vous recommandons d'utiliser des noms de variables uniques. Pour plus d'informations, consultez Transmission d'une variable d'environnement individuelle à un conteneur Amazon ECS.

`value`

Type : chaîne

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : chaîne

Obligatoire : oui

Type de fichier à utiliser La seule valeur prise en charge est `s3`.

`environment`

Type : tableau d'objets

Obligatoire : non

Variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `Env` dans la commande docker create-container et à l'option `--env` dans la commande docker run.

> **Important**
>
> Nous déconseillons l'utilisation des variables d'environnement en texte clair pour les informations sensibles, comme les données d'identification.

`name`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Valeur de la variable d'environnement.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Type : tableau d'objets

Obligatoire : non

Objet représentant le secret à exposer à votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer au conteneur. Les valeurs prises en charge sont soit le nom de ressource Amazon complet (ARN), soit AWS Secrets Manager secret ou l'ARN complet du paramètre dans le AWS Systems Manager Magasin de paramètres.

> **Note**
>
> Si le paramètre Systems Manager Parameter Store ou le paramètre Secrets Manager existe dans le même Région AWS comme tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Sécurité

`privileged`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, le conteneur dispose de droits élevés sur l'instance de conteneur hôte (similaire à l'utilisateur `root`). Ce paramètre correspond à `Privileged` dans la commande docker create-container et à l'option `--privileged` dans la commande docker run.

`user`

Type : chaîne

Obligatoire : non

Utilisateur à utiliser à l'intérieur du conteneur. Ce paramètre correspond à `User` dans la commande docker create-container et à l'option `--user` dans la commande docker run.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, vous ne devez pas exécuter de conteneurs à l'aide de l'utilisateur root (UID 0). Pour plus de sécurité, nous recommandons l'utilisation d'un utilisateur non root.

Vous pouvez spécifier l'`user` à l'aide des formats suivants. Si vous spécifiez un UID ou GID, vous devez le spécifier en tant que nombre entier positif.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à `ReadonlyRootfs` dans la commande docker create-container et à l'option `--read-only` dans la commande docker run.

`dockerSecurityOptions`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : tableau de chaînes

Obligatoire : non

Une liste de chaînes pour fournir des étiquettes personnalisées pour SELinux et AppArmor systèmes de sécurité à plusieurs niveaux. Ce champ n'est pas valide pour les conteneurs dans les tâches qui utilisent Fargate.

`ulimits`

Type : tableau d'objets Ulimit

Obligatoire : non

Une liste d'`ulimits` à définir dans le conteneur. Si une valeur ulimit est spécifiée dans une définition de tâche, elle remplace les valeurs par défaut définies par Docker. Ce paramètre correspond à `Ulimits` dans la commande docker create-container et à l'option `--ulimit` dans la commande docker run. Les valeurs d'attribution de nom valides sont affichées dans le type de données Ulimit.

Les tâches Amazon ECS hébergées sur Fargate utilisent les valeurs de limite définies par défaut par le système d'exploitation, à l'exception du paramètre de limite de ressources `nofile` qui est remplacé par Fargate. La limite de ressources `nofile` restreint le nombre de fichiers ouverts qu'un conteneur peut utiliser. Par défaut, la limite souple `nofile` a la valeur `1024` et la limite stricte a la valeur `65535`.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur. Pour vérifier la version de l'API Docker à distance de votre instance de conteneur, connectez-vous à votre instance de conteneur et exécutez la commande suivante : `sudo docker version --format '{{.Server.APIVersion}}'`

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

UNE key/value carte des étiquettes à ajouter au contenant. Ce paramètre correspond à `Labels` dans la commande docker create-container et à l'option `--label` dans la commande docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

```
"dockerLabels": {"string": "string"
      ...}
```

Paramètres réseau

`disableNetworking`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, la mise en réseau est désactivée dans le conteneur.

La valeur par défaut est `false`.

```
"disableNetworking": true|false
```

`links`

> **Note**
>
> Ce paramètre n’est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : tableau de chaînes

Obligatoire : non

Le paramètre `link` permet aux conteneurs de communiquer entre eux sans avoir besoin de définir des mappages de ports. Ce paramètre n'est pris en charge que si le mode réseau d'une définition de tâche est défini sur `bridge`. La construction `name:internalName` est analogue à `name:alias` dans les liaisons Docker. Il peut comporter jusqu’à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement.

> **Important**
>
> Des conteneurs situés sur la même instance de conteneur peuvent communiquer entre eux sans nécessiter de liens ou de mappages de ports hôtes. L'isolation réseau sur une instance de conteneur est contrôlée par les groupes de sécurité et les paramètres de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

> **Note**
>
> Ce paramètre n’est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : chaîne

Obligatoire : non

Nom d'hôte à utiliser pour votre conteneur. Ce paramètre correspond à `Hostname` dans docker create-container et à l'option `--hostname` de docker run.

```
"hostname": "string"
```

`dnsServers`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : tableau de chaînes

Obligatoire : non

Liste de serveurs DNS qui sont présentés au conteneur.

```
"dnsServers": ["string", ...]
```

`extraHosts`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches qui utilisent le mode réseau `awsvpc`.

Type : tableau d'objets

Obligatoire : non

Liste de noms d'hôte et de mappages d'adresses IP à ajouter au fichier `/etc/hosts` dans le conteneur.

Ce paramètre correspond à `ExtraHosts` dans la commande docker create-container et à l'option `--add-host` dans la commande docker run.

```
"extraHosts": [
```

```
    {
      "hostname": "string",
      "ipAddress": "string"
    }
    ...
  ]
```

`hostname`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Adresse IP à utiliser dans l'entrée `/etc/hosts`.

Stockage et journalisation

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à `ReadonlyRootfs` dans la commande docker create-container et à l'option `--read-only` dans la commande docker run.

La valeur par défaut est `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur des instances EC2 exécutant le système d'exploitation Windows, laissez la valeur `false` par défaut.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond à `VolumesFrom` dans la commande docker create-container et à l'option `--volumes-from` dans la commande docker run.

`sourceContainer`

Type : String

Obligatoire : oui lorsque `volumesFrom` est utilisé

Nom du conteneur à partir duquel monter les volumes.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tapez : LogConfiguration Objet

Obligatoire : non

Spécification de configuration du journal pour le conteneur.

Pour obtenir des exemples de définitions de tâche qui utilisent une configuration de journaux, consultez Exemple de définitions de tâches Amazon ECS.

Ce paramètre correspond à `LogConfig` dans la commande docker create-container et à l'option `--log-driver` dans la commande docker run. Par défaut, les conteneurs utilisent le même pilote de journalisation que celui utilisé par le démon Docker. Cependant, le conteneur peut utiliser un pilote de journalisation différent que celui du démon Docker en spécifiant un pilote de journal avec ce paramètre dans la définition de conteneur. Pour utiliser un pilote de journalisation différent pour un conteneur, le système de journal doit être configuré correctement sur l'instance de conteneur (ou sur un serveur de journal différent pour les options de journalisation à distance).

Tenez compte des éléments suivants lorsque vous spécifiez une configuration de journal pour vos conteneurs :

- Amazon ECS prend en charge un sous-ensemble des pilotes de journalisation qui sont disponibles pour le démon Docker.
- Ce paramètre nécessite la version 1.18 ou ultérieure de l'API Docker à distance sur votre instance de conteneur.

```
"logConfiguration": {
      "logDriver": "awslogs",""splunk", "awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : String

Valeurs valides : `"awslogs","splunk","awsfirelens"`

Obligatoire : oui lorsque `logConfiguration` est utilisé

Pilote de journal à utiliser pour le conteneur. Par défaut, les valeurs valides mentionnées ci-dessus sont les pilotes de journal avec lesquels l'agent de conteneur Amazon ECS peut communiquer.

Les pilotes de journalisation pris en charge sont `awslogs`, `splunk` et `awsfirelens`.

Pour plus d'informations sur l'utilisation du `awslogs` enregistrez le pilote dans les définitions de tâches pour envoyer les journaux de vos conteneurs à CloudWatch Journaux, voir Envoyer les journaux Amazon ECS à CloudWatch.

Pour plus d'informations sur l'utilisation du pilote de journalisation `awsfirelens`, consultez la section Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

> **Note**
>
> Si vous avez un pilote personnalisé qui n'est pas répertorié, vous pouvez bifurquer le projet d'agent de conteneur Amazon ECS qui est disponible sur GitHub et

personnalisez-le pour qu'il fonctionne avec ce pilote. Nous vous conseillons d'envoyer des demandes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, nous ne prenons pas en charge l'exécution de copies modifiées de ce logiciel pour le moment.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options de configuration à envoyer au pilote de journal.

Les options que vous pouvez spécifier dépendent du pilote de journalisation. Certaines des options que vous pouvez spécifier lorsque vous utilisez le `awslogs` routeur pour acheminer les journaux vers Amazon CloudWatch inclure les éléments suivants :

`awslogs-create-group`

Obligatoire : non

Spécifiez si vous souhaitez que le groupe de journaux soit créé automatiquement. Si cette option n'est pas spécifiée, l'emplacement par défaut est `false`.

> **Note**
>
> Votre politique IAM doit inclure l'autorisation `logs:CreateLogGroup` afin que vous puissiez essayer d'utiliser `awslogs-create-group`.

`awslogs-region`

Obligatoire : oui

Spécifiez le Région AWS que le `awslogs` log driver est destiné à envoyer vos journaux Docker à. Vous pouvez choisir d'envoyer tous vos journaux provenant de clusters de différentes régions vers une seule région dans CloudWatch Journaux. Cela leur permet

d'être tous visibles en un seul endroit. Sinon, vous pouvez les séparer par région pour plus de granularité. Assurez-vous que le groupe de journaux spécifié existe dans la région que vous spécifiez avec cette option.

`awslogs-group`

Obligatoire : oui

Assurez-vous de spécifier un groupe de journaux auquel le pilote de journaux `awslogs` envoie ses flux de journaux.

`awslogs-stream-prefix`

Obligatoire : oui

Utilisez l'option `awslogs-stream-prefix` pour associer un flux de journaux au préfixe spécifié, au nom du conteneur et à l'ID de la tâche Amazon ECS à laquelle appartient le conteneur. Si vous spécifiez un préfixe avec cette option, le format du flux de journaux est le suivant.

```
prefix-name/container-name/ecs-task-id
```

Si vous ne spécifiez pas de préfixe avec cette option, le flux de journaux est nommé d'après l'ID de conteneur attribué par le démon Docker sur l'instance de conteneur. Etant donné qu'il est difficile de suivre les journaux jusqu'au conteneur qui les a envoyé en utilisant uniquement l'ID de conteneur Docker (qui est uniquement disponible sur l'instance de conteneur), nous vous recommandons de spécifier un préfixe avec cette option.

Pour les services Amazon ECS, vous pouvez utiliser le nom du service comme préfixe. Ce faisant, vous pourrez suivre les flux de journaux et déterminer le service auquel appartient le conteneur, trouver le nom du conteneur qui les a envoyés, ainsi que l'ID de la tâche à laquelle le conteneur appartient.

Vous devez spécifier un préfixe de flux pour vos journaux pour qu'ils apparaissent dans le panneau des journaux lors de l'utilisation de la console Amazon ECS.

`awslogs-datetime-format`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes au format `strftime` Python. Un message de journal est composé de plusieurs lignes : une ligne correspondant

au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Ce format peut, par exemple, servir à analyser une sortie comme une pile de vidage, laquelle pourrait, dans le cas contraire, être consignée en plusieurs entrées. Le modèle adéquat permet de la capturer dans une seule entrée.

Pour plus d'informations, consultez awslogs-datetime-format.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`awslogs-multiline-pattern`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes à l'aide d'une expression régulière. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Pour plus d'informations, consultez awslogs-multiline-pattern.

Cette option est ignorée si `awslogs-datetime-format` est également configuré.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Cette option définit le mode de transmission des messages de journalisation du conteneur vers le pilote de journalisation `awslogs`. Le mode de livraison que vous choisissez affecte la disponibilité de l'application lorsque le flux des journaux provenant du conteneur est interrompu.

Si vous utilisez le `blocking` mode et flux de journaux vers CloudWatch est interrompu, appelle depuis le code du conteneur pour écrire dans le `stdout` et `stderr` les flux seront bloqués. Le fil de journalisation de l'application se bloquera en conséquence. Cela peut empêcher l'application de répondre et entraîner l'échec de vérification de l'état du conteneur.

Si vous utilisez le mode `non-blocking`, les journaux du conteneur sont plutôt stockés dans une mémoire tampon intermédiaire configurée avec l'option `max-buffer-size`. Cela empêche l'application de ne pas répondre lorsque les journaux ne peuvent pas être envoyés à CloudWatch. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir la disponibilité du service et si vous acceptez une certaine perte de journaux. Pour plus d'informations, consultez la section Prévention de la perte de journaux grâce au mode non bloquant dans le pilote de journalisation conteneur `awslogs`.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `1m`

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. Lorsque la mémoire tampon est pleine, il est impossible de stocker d'autres journaux. Les journaux qui ne peuvent pas être stockés sont perdus.

Pour acheminer les journaux à l'aide du routeur de journaux `splunk`, vous devez spécifier un `splunk-token` et une `splunk-url`.

Lorsque vous utilisez le `awsfirelens` routeur de journaux pour acheminer les journaux vers un Service AWS ou AWS Partner Network destination pour le stockage et l'analyse des

journaux, vous pouvez définir `log-driver-buffer-limit` option pour limiter le nombre d'événements qui sont mis en mémoire tampon avant d'être envoyés au conteneur Log Router. Cela peut aider à résoudre le problème potentiel de perte de journal, car un débit élevé peut entraîner un épuisement de la mémoire pour le tampon à l'intérieur de Docker. Pour plus d'informations, consultez the section called “Configuration des journaux pour un débit élevé”.

Les autres options que vous pouvez spécifier lors de l'utilisation d'`awsfirelens` pour acheminer les journaux dépendent de la destination. Lorsque vous exportez des journaux vers Amazon Data Firehose, vous pouvez spécifier Région AWS avec `region` et un nom pour le flux de journaux avec `delivery_stream`.

Lorsque vous exportez des journaux vers Amazon Kinesis Data Streams, vous pouvez spécifier la Région AWS avec `region` et un nom pour le flux de données avec `stream`.

Lorsque vous exportez des journaux vers Amazon OpenSearch Service, vous pouvez spécifier des options telles que `Name`, `Host` (OpenSearch Point de terminaison de service (sans protocole), `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name`, et `tls`.

Lorsque vous exportez des journaux vers Amazon S3, vous pouvez spécifier le compartiment à l'aide de l'option `bucket`. Vous pouvez également spécifier `region`, `total_file_size`, `upload_timeout` et `use_put_object` en tant qu'options.

Ce paramètre nécessite la version 1.19 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`secretOptions`

Type : tableau d'objets

Obligatoire : non

Objet qui représente le secret à transmettre à la configuration de journal. Les secrets utilisés dans la configuration du journal peuvent inclure un jeton d'authentification, un certificat ou une clé de chiffrement. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer à la configuration de journal du conteneur.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tapez : FirelensConfiguration Objet

Obligatoire : non

Le FireLens configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options à utiliser lors de la configuration du log router. Ce champ est facultatif et peut être utilisé pour spécifier un fichier de configuration personnalisé ou ajouter des métadonnées supplémentaires, telles que les détails de tâche, de définition de tâche, de cluster et d'instance de conteneur à l'événement de journal. Si spécifié, la syntaxe à utiliser est `"options":{"enable-ecs-log-metadata":"true|false","config-file-type":"file","config-file-value":"filepath"}`. Pour plus d'informations, consultez Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

`type`

Type : chaîne

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Besoins en ressources

`resourceRequirements`

Type : tableau d'objets ResourceRequirement

Obligatoire : non

Le type et la quantité de chaque ressource à affecter à un conteneur. Les ressources prises en charge sont les GPU et les appareils Neuron.

`type`

Type : chaîne

Obligatoire : oui

Type d'une ressource à attribuer à un conteneur. Les valeurs prises en charge sont `GPU` et `NeuronDevice`.

`value`

Type : chaîne

Obligatoire : oui

Valeur du type de ressource spécifié.

Si le type `GPU` est utilisé, la valeur est le nombre d'éléments `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Vous pouvez spécifier une valeur entière pour les GPU complets, une valeur décimale (`0.125`, `0.25`, ou `0.5`) pour les GPU fractionnaires sur les instances G6f, ou `ALL` pour allouer tous les GPU de l'instance de conteneur au conteneur. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne peut pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée.

Les GPU ne sont pas disponibles pour les tâches exécutées sur Fargate.

Si le `NeuronDevice` le type est utilisé, la seule valeur prise en charge est `ALL`. Quand `ALL` est spécifié, le conteneur obtient un accès exclusif à tous les appareils Neuron de l'instance.

Les appareils Neuron ne sont pas disponibles pour les tâches exécutées sur Fargate.

`sharingStrategy`

Type : objet SharingStrategy

Obligatoire : non

Configuration de partage du GPU pour le conteneur. Spécifiez `sharingStrategy` pour partager un seul GPU physique sur plusieurs conteneurs avec NVIDIA Multi-Process Service (MPS). Cet objet contient `nvidiaMps` objet avec les champs suivants :

Sur un `GPU` besoins en ressources, `value` et `sharingStrategy` s'excluent mutuellement. Lorsque vous spécifiez `sharingStrategy`, vous devez omettre `value`.

`memory`

Type : Integer

Obligatoire : oui

Limite de mémoire GPU pour le conteneur, en MiB. Cette valeur représente un plafond strict de la mémoire GPU que le conteneur peut utiliser. La valeur doit être supérieure à `0`.

`maxComputePercent`

Type : Integer

Obligatoire : non

Le pourcentage maximum de calcul GPU que le conteneur peut utiliser, sous la forme d'un entier de 1 pour `100`. Cette valeur est un plafond et non une réservation. Si vous l'omettez, le conteneur peut utiliser jusqu'à `100` pour cent.

Temporisations de conteneurs

`startTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée d'attente (en secondes) avant l'abandon de la résolution de dépendances pour un conteneur.

Par exemple, vous spécifiez deux conteneurs dans une définition de tâche, où le `containerA` dispose d'une dépendance au `containerB` avec un état `COMPLETE`, `SUCCESS` ou `HEALTHY`. Si une valeur `startTimeout` est spécifiée pour `containerB` et qu'il n'a pas atteint l'état souhaité dans ce délai, alors le `containerA` ne démarre pas.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

La valeur maximale est de 600 secondes (10 minutes).

`stopTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée (en secondes) à attendre avant que le conteneur soit expressément tué s'il ne s'achève pas normalement tout seul.

Si le paramètre n'est pas spécifié, alors la valeur par défaut de 30 secondes est utilisée. La valeur maximale est de 86 400 secondes (24 heures).

Dépendances du conteneur

`dependsOn`

Type : tableau d'objets ContainerDependency

Obligatoire : non

Dépendances définies pour le démarrage et l'arrêt de conteneurs. Un conteneur peut contenir plusieurs dépendances. Lorsqu'une dépendance est définie pour le démarrage des conteneurs, elle est inversée pour leur arrêt. Pour obtenir un exemple, consultez Dépendances du conteneur.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version ultérieure (Linux) ou `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Type : chaîne

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : chaîne

Obligatoire : oui

Condition de dépendance du conteneur. Voici les conditions disponibles et leur comportement :

- `START` - Cette condition émule le comportement de liens et de volumes aujourd'hui. La condition vérifie qu'un conteneur dépendant est démarré avant d'autoriser d'autres conteneurs à démarrer.
- `COMPLETE` - Cette condition vérifie qu'un conteneur dépendant exécute un cycle complet (sorties) avant d'autoriser d'autres conteneurs à démarrer. Cela peut être utile pour les conteneurs non essentiels qui exécutent un script, puis se ferment. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `SUCCESS` - Cette condition est identique à `COMPLETE`, mais elle nécessite également que le conteneur se termine par un état `zero`. Cette condition ne peut pas être définie sur un contenant essentiel.
- `HEALTHY` - Cette condition vérifie que la surveillance de l'état du conteneur dépendant réussit avant d'autoriser d'autres conteneurs à démarrer. Cela nécessite que le conteneur dépendant dispose de surveillances de l'état configurées dans la définition de la tâche. Cette condition n'est confirmée qu'au démarrage de la tâche.

Contrôles système

`systemControls`

Type : objet SystemControl

Obligatoire : non

Liste des paramètres du noyau de l’espace de noms à définir dans le conteneur. Ce paramètre correspond à `Sysctls` dans la commande docker create-container et à l’option `--sysctl` dans la commande docker run. Par exemple, vous pouvez configurer le paramètre `net.ipv4.tcp_keepalive_time` pour maintenir des connexions de plus longue durée.

Il n'est pas recommandé de spécifier des paramètres `systemControls` liés au réseau pour plusieurs conteneurs dans une seule tâche qui utilise également le mode réseau `awsvpc` ou `host`. En procédant ainsi, vous vous exposez aux inconvénients suivants :

- Si vous définissez `systemControls` pour n'importe quel conteneur, cela s'applique à tous les conteneurs de la tâche. Si vous définissez différents paramètres `systemControls` pour plusieurs conteneurs dans une seule tâche, c'est le conteneur qui est démarré en dernier qui détermine le paramètre `systemControls` qui prend effet.

Si vous définissez un espace de noms des ressources IPC à utiliser pour les conteneurs de la tâche, les conditions suivantes s'appliquent aux contrôles de système. Pour plus d'informations, consultez Mode IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les valeurs des `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

```
"systemControls": [
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Type : chaîne

Obligatoire : non

Paramètre du noyau de l'espace de noms pour lequel définir une `value`.

Valeurs d'espace de noms IPC valides : `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"`, et `Sysctls` qui commencent par `"fs.mqueue.*"`

Valeurs d'espace de noms de réseau valides : `Sysctls` qui commencent par `"net.*"` Sur Fargate, seuls les espaces de noms `Sysctls` existant dans le conteneur sont acceptés.

Toutes ces valeurs sont prises en charge par Fargate.

`value`

Type : chaîne

Obligatoire : non

Valeur du paramètre du noyau de l'espace de noms spécifié dans `namespace`.

Interactive

`interactive`

Type : booléen

Obligatoire : non

Lorsque ce paramètre est `true`, vous pouvez déployer des applications conteneurisées qui nécessitent l'allocation d'une `stdin` ou d'un `tty`. Ce paramètre correspond à `OpenStdin` dans la commande docker create-container et à l'option `--interactive` dans la commande docker run.

La valeur par défaut est `false`.

Pseudo Terminal

`pseudoTerminal`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, un TTY est alloué. Ce paramètre correspond à `Tty` dans la commande docker create-container et à l'option `--tty` dans la commande docker run.

La valeur par défaut est `false`.

Paramètres Linux

`linuxParameters`

Type : objet LinuxParameters

Obligatoire : non

Linux-specific les modifications appliquées au conteneur, telles que les fonctionnalités du noyau Linux.

`capabilities`

Type : objet KernelCapabilities

Obligatoire : non

Caractéristiques Linux du conteneur ajoutées ou supprimées de la configuration par défaut fournie par Docker.

`devices`

Type : tableau d'objets Périphérique

Obligatoire : non

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond à `Devices` dans la commande docker create-container et à l'option `--device` dans la commande docker run.

`initProcessEnabled`

Type : booléen

Obligatoire : non

Exécutez un processus `init` dans le conteneur afin de transmettre des signaux et de récolter les processus. Ce paramètre correspond à l'option `--init` de docker run.

`maxSwap`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : Integer

Obligatoire : non

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est converti en l'option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur et de la valeur `maxSwap`.

`swappiness`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : Integer

Obligatoire : non

Ce paramètre vous permet de régler le comportement d'échange de mémoire d'un conteneur. Si la valeur de `swappiness` est `0`, l'échange n'a pas lieu, sauf si cela s'avère absolument nécessaire. Avec la valeur `100` pour `swappiness`, l'échange de pages a lieu de manière très agressive. Les valeurs valides sont les nombres entiers compris entre `0` et `100`. Si le paramètre `swappiness` n'est pas spécifié, la valeur par défaut `60` est utilisée. Si aucune valeur n'est spécifiée pour `maxSwap`, le paramètre est ignoré. Ce paramètre correspond à l'option `--memory-swappiness` de docker run.

`sharedMemorySize`

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les tâches exécutées sur les instances gérées Amazon ECS.

Type : Integer

Obligatoire : non

Taille (en Mio) du volume `/dev/shm`. Ce paramètre correspond à l'option `--shm-size` de docker run.

`tmpfs`

Type : tableau d'objets Tmpfs

Obligatoire : non

Chemin du conteneur, options de montage et taille (en Mio) du montage tmpfs. Ce paramètre correspond à l'option `--tmpfs` de docker run.

Paramètres de définition de tâche Amazon ECS pour Fargate

Les définitions de tâches sont divisées en plusieurs parties : la famille de tâches, le Gestion des identités et des accès AWS le rôle des tâches (IAM), le mode réseau, les définitions de conteneurs, les volumes et les types de lancement. Les définitions de famille et de conteneur sont requises dans une définition de tâche. En revanche, le rôle de tâche, le mode réseau, les volumes et le type de lancement sont facultatifs.

Vous pouvez utiliser ces paramètres dans un fichier JSON pour configurer votre définition de tâche.

Voici des descriptions plus détaillées de chaque paramètre de définition de tâche pour Fargate.

Family

`family`

Type : chaîne

Obligatoire : oui

Lorsque vous enregistrez une définition de tâche, vous lui attribuez une famille. Cela équivaut, pour plusieurs versions de définition de tâche, à lui attribuer un nom spécifié avec un numéro de révision. La première définition de tâche enregistrée dans une famille donnée reçoit le numéro de révision 1. Toute définition de tâche enregistrée après celle-ci reçoit un numéro de révision ultérieur dans l'ordre séquentiel.

Capacity

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la capacité par rapport à laquelle Amazon ECS doit valider la définition de tâche. Une exception client est renvoyée si la définition de tâche n'est pas conforme aux compatibilités spécifiées.

Le paramètre suivant est autorisé dans une définition de tâche.

`requiresCompatibilities`

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `FARGATE`

La capacité par rapport à laquelle valider la définition de tâche. Cela permet de lancer une vérification afin de s'assurer que tous les paramètres utilisés dans la définition de la tâche correspondent aux exigences de Fargate.

Rôle de tâche

`taskRoleArn`

Type : chaîne

Obligatoire : non

Lorsque vous enregistrez une définition de tâche, vous pouvez fournir un rôle de tâche pour un rôle IAM qui permet aux conteneurs dans l'autorisation de tâche d'appeler en votre nom les API AWS qui sont spécifiées dans les stratégies qui lui associées. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Rôle d'exécution de tâche

`executionRoleArn`

Type : String

Obligatoire : Conditionnelle

Le nom de ressource Amazon (ARN) du rôle d'exécution de la tâche qui autorise l'agent de conteneur Amazon ECS à effectuer AWS Appels d'API en votre nom.

> **Note**
>
> Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Mode réseau

`networkMode`

Type : chaîne

Obligatoire : oui

Mode réseau Docker à utiliser pour les conteneurs de la tâche. Pour les tâches Amazon ECS hébergées sur Fargate, le mode réseau `awsvpc` est requis.

Plateforme d'exécution

`operatingSystemFamily`

Type : String

Obligatoire : Conditionnelle

Par défaut : LINUX

Ce paramètre est requis pour les tâches Amazon ECS hébergées sur Fargate.

Lorsque vous enregistrez une définition de tâche, vous spécifiez la famille du système d'exploitation.

Les valeurs valides sont `LINUX`, `WINDOWS_SERVER_2025_FULL`, `WINDOWS_SERVER_2025_CORE`, `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_2019_FULL` et `WINDOWS_SERVER_2019_CORE`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsqu'une définition de tâche fait partie d'un service, cette valeur doit correspondre à la valeur `platformFamily` du service.

`cpuArchitecture`

Type : String

Obligatoire : Conditionnelle

Par défaut : X86_64

Si le paramètre est laissé sur `null`, la valeur par défaut est automatiquement attribuée lors du lancement d'une tâche hébergée sur Fargate.

Lorsque vous enregistrez une définition de tâche, vous spécifiez l'architecture du processeur. Les valeurs valides sont `X86_64` et `ARM64`.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsque vous avez des tâches Linux, vous pouvez définir la valeur sur `ARM64`. Pour plus d'informations, consultez the section called “Définitions de tâche pour les charges de travail ARM 64 bits”.

Taille de la tâche

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la quantité totale d'UC et de mémoire utilisée pour cette tâche. Ces valeurs sont distinctes des valeurs `cpu` et `memory` au niveau de la définition de conteneur. Pour les tâches hébergées sur Fargate (Linux et Windows), ces champs sont obligatoires et des valeurs spécifiques sont disponibles selon le `cpu` et la `memory` pris en charge.

Le paramètre suivant est autorisé dans une définition de tâche :

`cpu`

Type : chaîne

Obligatoire : oui

> **Note**
>
> Task-level Les paramètres du processeur et de la mémoire sont requis et utilisés pour déterminer le type d'instance et la taille sur lesquels les tâches s'exécutent. Pour les tâches Windows, ces valeurs ne sont pas imposées au moment de l'exécution, car Windows ne dispose pas d'un mécanisme natif permettant d'appliquer facilement des limites de ressources collectives à un groupe de conteneurs. Si vous souhaitez imposer

des limites de ressources, nous vous recommandons d'utiliser les ressources au niveau du conteneur pour les conteneurs Windows.

Limite stricte du nombre d'unités UC à présenter pour la tâche. Vous pouvez spécifier les valeurs de l'UC dans le fichier JSON sous forme de chaîne en unités d'UC ou en processeurs virtuels (vCPU). Par exemple, vous pouvez spécifier une valeur d'UC de `1024` en unités d'UC ou de `1 vCPU` en vCPU. Lorsque la définition de tâche est enregistrée, la valeur de vCPU (processeurs virtuels) est convertie en un nombre entier indiquant les unités UC.

Ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine la plage de valeurs prises en charge pour le paramètre `memory`. Le tableau suivant indique les combinaisons valides d'UC et de mémoire au niveau de la tâche.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
16384 (16 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux
32768 (32 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	60 GO, 120 GO, 244 GO	Linux

`memory`

Type : chaîne

Obligatoire : oui

Note

Task-level Les paramètres du processeur et de la mémoire sont requis et utilisés pour déterminer le type d'instance et la taille sur lesquels les tâches s'exécutent. Pour les tâches Windows, ces valeurs ne sont pas imposées au moment de l'exécution, car Windows ne dispose pas d'un mécanisme natif permettant d'appliquer facilement des limites de ressources collectives à un groupe de conteneurs. Si vous souhaitez imposer

des limites de ressources, nous vous recommandons d'utiliser les ressources au niveau du conteneur pour les conteneurs Windows.

Limite stricte de quantité de mémoire à allouer à la tâche. Vous pouvez spécifier les valeurs de mémoire dans la définition de la tâche sous forme de chaîne en mébioctets (Mio) ou en gigaoctets (Go). Par exemple, vous pouvez spécifier une valeur de mémoire de `3072` en Mio ou de `3 GB` en Go. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Ce champ est obligatoire et vous devez utiliser l'une des valeurs suivantes, qui détermine la plage de valeurs prises en charge pour le paramètre `cpu` :

Valeur de mémoire (en Mio, avec une valeur équivalente approximative en Go)	Valeur d'UC	Systèmes d'exploitation pris en charge pour Fargate
512 (0,5 Go), 1 024 (1 Go), 2 048 (2 Go)	256 (0,25 vCPU)	Linux
1 024 (1 Go), 2 048 (2 Go), 3 072 (3 Go), 4 096 (4 Go)	512 (0,5 vCPU)	Linux
2 048 (2 Go), 3072 (3 Go), 4 096 (4 Go), 5 120 (5 Go), 6 144 (6 Go), 7 168 (7 Go), 8 192 (8 Go)	1 024 (1 vCPU)	Linux, Windows
Entre 4 096 (4 Go) et 16 384 (16 Go) par incréments de 1 024 (1 Go)	2 048 (2 vCPU)	Linux, Windows
Entre 8 192 (8 Go) et 30 720 (30 Go) par incréments de 1 024 (1 Go)	4 096 (4 vCPU)	Linux, Windows
Entre 16 Go et 60 Go par incréments de 4 Go	8192 (8 vCPU)	Linux

Valeur de mémoire (en Mio, avec une valeur équivalente approximative en Go)	Valeur d'UC	Systèmes d'exploitation pris en charge pour Fargate
Note Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure		
Entre 32 Go et 120 Go par incréments de 8 Go **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	16384 (16vCPU)	Linux

Définitions de conteneur

Lorsque vous enregistrez une définition de tâche, vous devez spécifier une liste de définitions de conteneur qui sont transmises au démon Docker sur une instance de conteneur. Les paramètres suivants sont autorisés dans une définition de conteneur.

Rubriques

- Paramètres de définition de conteneur standards
- Paramètres de définition de conteneur avancés
- Autres paramètres de définition de conteneur

Paramètres de définition de conteneur standards

Les paramètres de définition de tâche suivants sont obligatoires ou utilisés dans la plupart des définitions de conteneur.

Rubriques

- Nom
- Image
- Mémoire
- Cartographie des ports
- Informations d'identification du référentiel privé

Nom

`name`

Type : chaîne

Obligatoire : oui

Le nom d'un conteneur. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés. Si vous associez plusieurs conteneurs dans une définition de tâche, vous pouvez spécifier l'option `name` d'un conteneur dans l'option `links` d'un autre conteneur. Cela permet de connecter les conteneurs.

Image

`image`

Type : chaîne

Obligatoire : oui

Image utilisée pour démarrer un conteneur. Cette chaîne est transmise directement au démon Docker. Par défaut, les images dans le registre Docker Hub sont disponibles. Vous pouvez également spécifier d'autres référentiels avec soit *`repository-url`*`/`*`image`*`:`*`tag`*, soit *`repository-url`*`/`*`image`*`@`*`digest`*. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, deux points, des points, des

barres obliques et des signes dièse. Ce paramètre correspond à `Image` dans la commande docker create-container et `IMAGE` paramètre de la commande Docker run.

- Lorsqu'une nouvelle tâche démarre, l'agent de conteneur Amazon ECS extrait la version la plus récente de l'image et de l'étiquette spécifiées afin que le conteneur puisse les utiliser. Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Lorsque vous ne spécifiez pas de balise ou de résumé dans le chemin de l'image dans la définition de la tâche, l'agent de conteneur Amazon ECS utilise le `latest` balise pour extraire l'image spécifiée.
- Les mises à jour ultérieures apportées à une image de référentiel ne sont pas propagées aux tâches déjà en cours d'exécution.
- Les images des registres privés sont prises en charge. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.
- Les images des référentiels Amazon ECR peuvent être spécifiées en utilisant soit la convention de dénomination complète `registry/repository:tag` ou `registry/repository@digest` (par exemple, *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`:`*`latest`* ou *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`my-web-app`*`@`*`sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE`*).
- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Type : String

Valeurs valides : `enabled|disabled`

Obligatoire : non

Spécifie si Amazon ECS résoudra la balise d'image de conteneur fournie dans la définition du conteneur en un condensé d'image. Par défaut, ce comportement est `enabled`. Si vous définissez la valeur d'un conteneur sur `disabled`, Amazon ECS ne résoudra pas la balise

d'image de conteneur en un condensé et utilisera l'URI d'image d'origine spécifiée dans la définition du conteneur pour le déploiement. Pour plus d'informations sur la résolution des images de conteneur, consultez la section Résolution des images de conteneurs.

Mémoire

`memory`

Type : Integer

Obligatoire : non

La quantité de mémoire (en Mio) à présenter au conteneur. Si votre conteneur tente de dépasser la mémoire spécifiée ici, il sera désactivé. La quantité totale de mémoire réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `memory` de la tâche, si cette valeur est spécifiée. Ce paramètre correspond à `Memory` dans la commande docker create-container et à l'option `--memory` dans la commande docker run.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

`memoryReservation`

Type : Integer

Obligatoire : non

La limite flexible (en Mio) de mémoire à réserver pour le conteneur. En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite

flexible. Toutefois, votre conteneur peut utiliser davantage de mémoire en cas de besoin. Le conteneur peut consommer jusqu'à la limite stricte spécifiée avec le paramètre `memory` (le cas échéant), ou la totalité de la mémoire disponible sur l'instance de conteneur, selon la première valeur atteinte. Ce paramètre correspond à `MemoryReservation` dans la commande docker create-container et à l'option `--memory-reservation` dans la commande docker run.

Si aucune valeur de mémoire au niveau de la tâche n'est spécifiée, vous devez indiquer un nombre entier différent de zéro pour `memory` ou `memoryReservation` dans une définition de conteneur. Si vous spécifiez les deux, `memory` doit être supérieur à `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Par exemple, supposons que votre conteneur utilise normalement 128 Mio de mémoire, mais qu'il lui arrive d'utiliser jusqu'à 256 Mio de mémoire pendant de courtes périodes. Vous pouvez définir une `memoryReservation` de 128 Mio et une limite stricte `memory` de 300 Mio. Cette configuration permet au conteneur de ne réserver que 128 Mio de mémoire à partir des ressources restantes sur l'instance de conteneur. En même temps, cette configuration permet également au conteneur d'utiliser davantage de ressources mémoire lorsque cela est nécessaire.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

Cartographie des ports

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports exposent les ports réseau de votre conteneur au monde extérieur, ce qui permet aux clients d'accéder à votre application. Ils sont également utilisés pour la communication entre conteneurs au sein d'une même tâche.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le `hostPort` est toujours ignoré et le port du conteneur est automatiquement mappé à un port aléatoire portant un numéro élevé sur l'hôte.

Les mappages de ports sur Windows utilisent l'adresse de passerelle `NetNAT` plutôt que `localhost`. Il n'existe aucune boucle pour les mappages de ports sous Windows, donc vous ne pouvez pas accéder au port mappé d'un conteneur à partir de l'hôte lui-même.

La plupart des champs de ce paramètre (y compris `containerPort`, `hostPort`, `protocol`) carte vers `PortBindings` dans la commande docker create-container et `--publish` option pour exécuter Docker. Si le mode réseau d'une définition de tâche est défini sur `host`, les ports hôtes doivent être indéfinis ou correspondre au port du conteneur dans le mappage de port.

> **Note**
>
> Une fois qu'une tâche passe à l'état `RUNNING`, les affectations manuelles et automatiques de ports de conteneur et d'hôte sont visibles aux emplacements suivants :
>
> - Console : la section Network Bindings (Liaisons réseau) de la description d'un conteneur pour une tâche sélectionnée.
> - AWS CLI : la section `networkBindings` de la sortie de la commande describe-tasks.
> - API : la réponse `DescribeTasks`.
> - Métadonnées : point de terminaison des métadonnées de la tâche.

`appProtocol`

Type : chaîne

Obligatoire : non

Protocole d'application utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Nous vous conseillons de définir ce paramètre de manière cohérente avec le protocole que votre application utilise. Si vous définissez ce paramètre, Amazon ECS ajoute une gestion de connexion spécifique au protocole au proxy Service Connect. Si vous définissez ce paramètre, Amazon ECS ajoute une télémétrie spécifique au protocole dans la console Amazon ECS et CloudWatch.

Si vous ne définissez aucune valeur pour ce paramètre, le protocole TCP est utilisé. Toutefois, Amazon ECS n'ajoute pas de télémétrie spécifique au protocole pour TCP.

Pour plus d'informations, consultez the section called " Connexion au service".

Valeurs de protocole valides : `"http" | "http2" | "grpc"`

`containerPort`

Type : Integer

Obligatoire : oui, lorsque des objets `portMappings` sont utilisés

Le numéro de port sur le conteneur qui est lié au port hôte spécifié par l'utilisateur ou affecté automatiquement.

Pour les tâches utilisant le mode réseau `awsvpc`, vous utilisez `containerPort` pour spécifier les ports exposés.

Pour les conteneurs Windows sur Fargate, vous ne pouvez pas utiliser le port 3 150 pour `containerPort`. C'est parce qu'il est réservé.

`containerPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement.

Vous ne pouvez définir ce paramètre qu'à l'aide de l'API `register-task-definition`. L'option est disponible dans le paramètre `portMappings`. Pour plus d'informations, consultez register-task-definition dans la Référence AWS Command Line Interface .

Les règles suivantes s'appliquent lorsque vous spécifiez une `containerPortRange` :

- Vous devez utiliser le mode réseau `awsvpc`.
- Ce paramètre est disponible pour les systèmes d'exploitation Windows et Linux.
- L'instance de conteneur doit au moins disposer de la version 1.67.0 de l'agent de conteneur et au moins de la version 1.67.0-1 du package `ecs-init`.
- Vous pouvez spécifier un maximum de 100 plages de ports pour chaque conteneur.
- Vous ne spécifiez pas de `hostPortRange`. La valeur de `hostPortRange` est définie comme suit :
  - Pour les conteneurs d'une tâche en mode réseau `awsvpc`, le `hostPort` est défini sur la même valeur que le `containerPort`. Il s'agit d'une stratégie de mappage statique.
- Les valeurs valides de `containerPortRange` sont comprises entre 1 et 65535.
- Un port peut uniquement être inclus dans un mappage de port pour chaque conteneur.
- Vous ne pouvez pas spécifier de plages de ports qui se chevauchent.
- Le premier port de la plage doit être inférieur au dernier port de la plage.
- Docker vous recommande de désactiver le Docker-proxy dans le Docker fichier de configuration du démon lorsque vous avez un grand nombre de ports.

  Pour plus d'informations, voir Numéro #11185 sur GitHub.

  Pour plus d'informations sur la désactivation du proxy Docker dans le fichier de configuration du démon Docker, veuillez consulter Démon Docker dans le Guide du développeur Amazon ECS (langue française non garantie).

Vous pouvez appeler DescribeTasks pour voir la `hostPortRange` correspondant aux ports hôtes liés aux ports de conteneur.

Les plages de ports ne sont pas incluses dans les événements de tâches Amazon ECS, qui sont envoyés à EventBridge. Pour plus d'informations, consultez the section called “Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge”.

`hostPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port sur l'hôte utilisée avec la liaison réseau. Elle est attribuée par Docker et délivrée par l'agent Amazon ECS.

`hostPort`

Type : Integer

Obligatoire : non

Le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Le `hostPort` peut rester vide ou avoir la même valeur que `containerPort`.

La plage de ports éphémères par défaut pour Docker 1.6.0 et versions ultérieures est répertoriée dans l'instance sous `/proc/sys/net/ipv4/ip_local_port_range`. Si ce paramètre de noyau n'est pas disponible, la plage de ports éphémères par défaut de `49153-65535` est utilisée. N'essayez pas de spécifier un port d'hôte dans la plage de ports éphémères. Cela est dû au fait qu'ils sont réservés à une affectation automatique. En général, les ports inférieurs à `32768` ne sont pas compris dans la plage de ports éphémères.

Les ports réservés par défaut sont le port `22` pour SSH, les ports Docker `2375` et `2376`, et les ports d'agent de conteneur Amazon ECS `51678-51680`. Tout port hôte ayant été spécifié précédemment par l'utilisateur pour une tâche en cours d'exécution est également réservé pendant l'exécution de la tâche. Après l'arrêt d'une tâche, le port hôte est libéré. Les ports réservés actuels s'affichent dans le champ `remainingResources` de la sortie describe-container-instances. Une instance de conteneur peut avoir jusqu'à 100 ports réservés à la fois, y compris les ports réservés par défaut. Les ports affectés automatiquement ne sont pas pris en compte dans le quota des 100 ports réservés.

`name`

Type : String

Obligatoire : Non, requis pour que Service Connect et VPC Lattice soient configurés dans un service

Nom utilisé pour le mappage de port. Ce paramètre ne s’applique qu’à Service Connect et VPC Lattice. Ce paramètre correspond au nom que vous utilisez dans la configuration Service Connect et VPC Lattice d’un service.

Pour plus d’informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.

Dans l’exemple suivant, les deux champs obligatoires pour Service Connect et VPC Lattice sont utilisés.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Type : chaîne

Obligatoire : non

Le protocole utilisé pour le mappage de port. Les valeurs valides sont `tcp` et `udp` (sensible aux majuscules et minuscules). La valeur par défaut est `tcp`. Amazon Amazon ECS traite toute autre valeur spécifiée comme `tcp`.

> **Important**
>
> Seul `tcp` est pris en charge pour Service Connect. N'oubliez pas que `tcp` est implicite si ce champ n'est pas défini.

Si vous spécifiez un port hôte, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
]
```

Si vous souhaitez utiliser un port hôte affecté automatiquement, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer
```

```
    }
    ...
]
```

Informations d'identification du référentiel privé

`repositoryCredentials`

Type : objet RepositoryCredentials

Obligatoire : non

Informations d'identification du référentiel pour l'authentification de registre privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

`credentialsParameter`

Type : String

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS, AWS CLI, ou AWS SDK, si le secret existe dans la même région que la tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Lorsque vous utilisez Console de gestion AWS, vous devez spécifier l'ARN complet du secret.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Paramètres de définition de conteneur avancés

Les paramètres avancés de définition de conteneur suivants fournissent des fonctionnalités étendues à la commande Docker run qui est utilisée pour lancer des conteneurs sur vos instances de conteneurs Amazon ECS.

Rubriques

- Politique de redémarrage
- Surveillance de l'état
- Environnement
- Paramètres réseau
- Stockage et journalisation
- Sécurité
- Limites des ressources
- Étiquettes Docker

Politique de redémarrage

`restartPolicy`

La politique de redémarrage du conteneur et les paramètres de configuration associés. Lorsque vous configurez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur sans avoir à remplacer la tâche. Pour plus d'informations, consultez Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`enabled`

Type : Boolean

Obligatoire : oui

Spécifie si une politique de redémarrage est activée pour le conteneur.

`ignoredExitCodes`

Type : tableau integer

Obligatoire : non

Liste des codes de sortie qu'Amazon ECS ignorera et pour lesquels il ne tentera pas de redémarrage. Vous pouvez spécifier un maximum de 50 codes de sortie de conteneur. Par défaut, Amazon ECS n'ignore aucun code de sortie.

`restartAttemptPeriod`

Type : Integer

Obligatoire : non

Durée (en secondes) pendant laquelle le conteneur doit fonctionner avant de pouvoir tenter un redémarrage. Un conteneur ne peut être redémarré qu'une fois toutes les `restartAttemptPeriod` secondes. Si un conteneur n'est pas en mesure de fonctionner pendant cette période et qu'il se ferme prématurément, il ne sera pas redémarré.
Vous pouvez définir une `restartAttemptPeriod` minimale de 60 secondes et une `restartAttemptPeriod` maximale de 1 800 secondes. Par défaut, un conteneur doit fonctionner pendant 300 secondes avant de pouvoir être redémarré.

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour plus d'informations, consultez Détermination de l'état des tâches Amazon ECS à l'aide de la surveillance de l'état des conteneurs.

`command`

Tableau de chaînes représentant la commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement

les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé.

Lors de l'enregistrement d'une définition de tâche dans le Console de gestion AWS, utilisez une liste de commandes séparées par des virgules. Ces commandes sont converties en une chaîne une fois la définition de tâche créée. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Lors de l'enregistrement d'une définition de tâche à l'aide du Console de gestion AWS panneau JSON, le AWS CLI, ou les API, placez la liste des commandes entre crochets. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un code de sortie de 0, avec aucune sortie `stderr`, indique la réussite et un code de sortie autre que zéro indique qu'il s'agit d'un échec.

`interval`

La durée (en secondes) entre chaque surveillance de l'état. Vous pouvez indiquer une durée comprise entre 5 et 300 secondes. La valeur par défaut est de 30 secondes.

`timeout`

La durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant qu'elle ne soit considérée comme un échec. Vous pouvez indiquer une durée comprise entre 2 et 60 secondes. La valeur par défaut est de 5 secondes.

`retries`

Nombre de nouvelles tentatives d'exécution d'une surveillance de l'état ayant échoué avant que le conteneur soit considéré comme défectueux. Vous pouvez indiquer un nombre de tentatives compris en 1 et 10. La valeur par défaut est de trois tentatives.

`startPeriod`

La période de grâce facultative pour donner aux conteneurs le temps de démarrer avant l'échec des surveillance de l'état est prise en compte dans le nombre maximal de tentatives. Vous pouvez spécifier une valeur comprise entre 0 et 300. Par défaut, la `startPeriod` est désactivée.

Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Environnement

`cpu`

Type : Integer

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sous Linux, ce paramètre correspond à `CpuShares` dans la section Créer un conteneur.

Ce champ est facultatif pour les tâches qui utilisent Fargate. La quantité totale de processeurs réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `cpu` au niveau de la tâche.

Les conteneurs Linux partagent des unités de processeur non allouées avec les autres conteneurs de l'instance de conteneur selon le même ratio que la quantité allouée. Par exemple, supposons que vous exécutez une tâche à conteneur unique sur un type d'instance à cœur unique avec 512 unités de processeur spécifiées pour ce conteneur. De plus, cette tâche est la seule tâche en cours d'exécution sur l'instance de conteneur. Dans cet exemple, le conteneur peut utiliser le partage complet de 1 024 unités UC à tout moment. Toutefois, supposons que vous ayez lancé une autre copie de la même tâche sur cette instance de conteneur. Un minimum de 512 unités de processeur est affecté à chaque tâche si nécessaire. De même, si l'autre conteneur n'utilise pas le processeur restant, chaque conteneur peut bénéficier d'une utilisation plus importante du processeur. Toutefois, si les deux tâches sont actives à 100 % en permanence, elles sont limitées à 512 unités d'UC.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur de processeur pour calculer les ratios de partage de processeur relatifs pour les conteneurs en cours d'exécution. La valeur de parts d’UC minimale autorisée par le noyau Linux est de 2, et la valeur de parts d’UC maximale autorisée par le noyau Linux est de 262 144. Toutefois, le paramètre UC n’est pas obligatoire et vous pouvez utiliser des valeurs d’UC inférieures à 2 ou supérieures à 262 144 dans vos définitions de conteneur. Pour les valeurs d’UC inférieures à 2 (y compris null) ou supérieures à 262 144, le comportement varie selon la version de l’agent de conteneur Amazon ECS :

Sur les instances de conteneur Windows, le quota de processeur est appliqué comme quota absolu. Les conteneurs Windows n'ont accès qu'à la quantité spécifiée de processeur décrite dans la définition de tâche. Une valeur de processeur nulle ou égale à zéro est transmise à Docker comme étant `0`. Windows interprète ensuite cette valeur comme 1 % d'un processeur.

Pour voir d'autres exemples, veuillez consulter le billet de blog How Amazon ECS manages CPU and memory resources.

`gpu`

Ce paramètre n’est pas pris en charge pour les conteneurs hébergés sur Fargate.

Type : objet ResourceRequirement

Obligatoire : non

Le nombre de `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne doit pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d’informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.

`Elastic Inference accelerator`

Ce paramètre n’est pas pris en charge pour les conteneurs hébergés sur Fargate.

Type : objet ResourceRequirement

Obligatoire : non

Pour le type `InferenceAccelerator`, `value` correspond à `deviceName` d'un `InferenceAccelerator` spécifié dans une définition de tâche. Pour plus d’informations, consultez the section called “Nom de l'accélérateur Elastic Inference (obsolète)”.

`essential`

Type : booléen

Obligatoire : non

Supposons que le paramètre `essential` d'un conteneur soit marqué comme `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque. Ensuite, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur a la valeur `false`,

son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Supposons que vous disposiez d'une application composée de plusieurs conteneurs. Ensuite, les conteneurs de groupes qui sont utilisés dans un même but pour former des composants, et séparer les différents composants en plusieurs définitions de tâches. Pour plus d'informations, consultez Architecture de votre application pour Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Les premières versions de l'agent de conteneur Amazon ECS ne traitent pas correctement les paramètres `entryPoint`. Si vous rencontrez des difficultés pour utiliser `entryPoint`, mettez à jour l'agent de conteneur ou saisissez plutôt vos commandes et vos arguments sous forme d'éléments de tableau `command`.

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond à `Cmd` dans la commande create-container et au paramètre `COMMAND` de docker run. S'il existe plusieurs arguments, assurez-vous que chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--workdir` correspond à docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Il n’est pas disponible pour les conteneurs Windows sur Fargate.

Type : tableau d'objets

Obligatoire : non

Liste des fichiers contenant les variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `--env-file` option de la commande Docker run.

Vous pouvez spécifier jusqu'à dix fichiers d'environnement. Le fichier doit avoir une extension de fichier `.env`. Chaque ligne d'un fichier d'environnement contient une variable d'environnement au format `VARIABLE=VALUE`. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées.

Si des variables d'environnement individuelles sont spécifiées dans la définition du conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement. Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités de haut en bas. Nous vous recommandons d'utiliser des noms de variables uniques. Pour plus d’informations, consultez Transmission d’une variable d’environnement individuelle à un conteneur Amazon ECS.

`value`

Type : chaîne

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : chaîne

Obligatoire : oui

Type de fichier à utiliser La seule valeur prise en charge est `s3`.

`environment`

Type : tableau d'objets

Obligatoire : non

Variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `Env` dans la commande docker create-container et à l'option `--env` dans la commande docker run.

> **Important**
>
> Nous déconseillons l'utilisation des variables d'environnement en texte clair pour les informations sensibles, comme les données d'identification.

`name`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Valeur de la variable d'environnement.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
```

```
]
```

`secrets`

Type : tableau d'objets

Obligatoire : non

Objet représentant le secret à exposer à votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer au conteneur. Les valeurs prises en charge sont soit le nom de ressource Amazon complet (ARN), soit AWS Secrets Manager secret ou l'ARN complet du paramètre dans le AWS Systems Manager Magasin de paramètres.

> **Note**
>
> Si le paramètre Systems Manager Parameter Store ou le paramètre Secrets Manager existe dans le même Région AWS comme tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
```

```
    }
]
```

Paramètres réseau

`disableNetworking`

Ce paramètre n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, la mise en réseau est désactivée dans le conteneur.

La valeur par défaut est `false`.

```
"disableNetworking": true|false
```

`links`

Ce paramètre n'est pas pris en charge pour les tâches qui utilisent le mode réseau `awsvpc`.

Type : tableau de chaînes

Obligatoire : non

Le paramètre `link` permet aux conteneurs de communiquer entre eux sans avoir besoin de définir des mappages de ports. Ce paramètre n'est pris en charge que si le mode réseau d'une définition de tâche est défini sur `bridge`. La construction `name:internalName` est analogue à `name:alias` dans les liaisons Docker. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement.

> **Important**
>
> Des conteneurs situés sur la même instance de conteneur peuvent communiquer entre eux sans nécessiter de liens ou de mappages de ports hôtes. L'isolation réseau sur une instance de conteneur est contrôlée par les groupes de sécurité et les paramètres de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Type : chaîne

Obligatoire : non

Nom d'hôte à utiliser pour votre conteneur. Ce paramètre correspond à `Hostname` dans docker create-container et à l'option `--hostname` de docker run.

> **Note**
>
> Si vous utilisez le mode réseau `awsvpc`, le paramètre `hostname` n'est pas pris en charge.

```
"hostname": "string"
```

`dnsServers`

Il n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : tableau de chaînes

Obligatoire : non

Liste de serveurs DNS qui sont présentés au conteneur.

```
"dnsServers": ["string", ...]
```

`extraHosts`

Ce paramètre n'est pas pris en charge pour les tâches qui utilisent le mode réseau `awsvpc`.

Type : tableau d'objets

Obligatoire : non

Liste de noms d'hôte et de mappages d'adresses IP à ajouter au fichier `/etc/hosts` dans le conteneur.

Ce paramètre correspond à `ExtraHosts` dans la commande docker create-container et à l'option `--add-host` dans la commande docker run.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
    ]
```

`hostname`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Adresse IP à utiliser dans l'entrée `/etc/hosts`.

Stockage et journalisation

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à `ReadonlyRootfs` dans la commande docker create-container et à l'option `--read-only` dans la commande docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

La valeur par défaut est `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches exécutées sur des instances EC2 exécutant le système d'exploitation Windows, laissez la valeur `false` par défaut.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond à `VolumesFrom` dans la commande docker create-container et à l'option `--volumes-from` dans la commande docker run.

`sourceContainer`

Type : String

Obligatoire : oui lorsque `volumesFrom` est utilisé

Nom du conteneur à partir duquel monter les volumes.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tapez : LogConfiguration Objet

Obligatoire : non

Spécification de configuration du journal pour le conteneur.

Pour obtenir des exemples de définitions de tâche qui utilisent une configuration de journaux, consultez Exemple de définitions de tâches Amazon ECS.

Ce paramètre correspond à `LogConfig` dans la commande docker create-container et à l'option `--log-driver` dans la commande docker run. Par défaut, les conteneurs utilisent le même pilote de journalisation que celui utilisé par le démon Docker. Cependant, le conteneur peut utiliser un pilote de journalisation différent que celui du démon Docker en spécifiant un pilote de journal avec ce paramètre dans la définition de conteneur. Pour utiliser un pilote de journalisation différent pour un conteneur, le système de journal doit être configuré correctement sur l'instance de conteneur (ou sur un serveur de journal différent pour les options de journalisation à distance).

Tenez compte des éléments suivants lorsque vous spécifiez une configuration de journal pour vos conteneurs :

- Amazon ECS prend en charge un sous-ensemble des pilotes de journalisation qui sont disponibles pour le démon Docker.
- Ce paramètre nécessite la version 1.18 ou ultérieure de l'API Docker à distance sur votre instance de conteneur.
- Vous devez installer tout logiciel supplémentaire en dehors de cette tâche. Par exemple, les agrégateurs de sortie Fluentd ou un hôte distant exécutant Logstash auquel envoyer des journaux Gelf.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : String

Valeurs valides : `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatoire : oui lorsque `logConfiguration` est utilisé

Pilote de journal à utiliser pour le conteneur. Par défaut, les valeurs valides mentionnées ci-dessus sont les pilotes de journal avec lesquels l'agent de conteneur Amazon ECS peut communiquer.

Les pilotes de journalisation pris en charge sont `awslogs`, `splunk` et `awsfirelens`.

Pour plus d'informations sur l'utilisation du `awslogs` enregistrez le pilote dans les définitions de tâches pour envoyer les journaux de vos conteneurs à CloudWatch Journaux, voir Envoyer les journaux Amazon ECS à CloudWatch.

Pour de plus amples informations sur l'utilisation du pilote de journal `awsfirelens`, consultez Routage personnalisé des journaux.

> **Note**
>
> Si vous avez un pilote personnalisé qui n'est pas répertorié, vous pouvez bifurquer le projet d'agent de conteneur Amazon ECS qui est disponible sur GitHub et personnalisez-le pour qu'il fonctionne avec ce pilote. Nous vous conseillons d'envoyer des demandes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, nous ne prenons pas en charge l'exécution de copies modifiées de ce logiciel pour le moment.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options de configuration à envoyer au pilote de journal.

Les options que vous pouvez spécifier dépendent du pilote de journalisation. Certaines des options que vous pouvez spécifier lorsque vous utilisez le `awslogs` routeur pour acheminer les journaux vers Amazon CloudWatch inclure les éléments suivants :

`awslogs-create-group`

Obligatoire : non

Spécifiez si vous souhaitez que le groupe de journaux soit créé automatiquement. Si cette option n'est pas spécifiée, l'emplacement par défaut est `false`.

> **Note**
>
> Votre politique IAM doit inclure l'autorisation `logs:CreateLogGroup` afin que vous puissiez essayer d'utiliser `awslogs-create-group`.

`awslogs-region`

Obligatoire : oui

Spécifiez le Région AWS que le `awslogs` log driver est destiné à envoyer vos journaux Docker à. Vous pouvez choisir d'envoyer tous vos journaux provenant de clusters de différentes régions vers une seule région dans CloudWatch Journaux. Cela leur permet d'être tous visibles en un seul endroit. Sinon, vous pouvez les séparer par région pour plus de granularité. Assurez-vous que le groupe de journaux spécifié existe dans la région que vous spécifiez avec cette option.

`awslogs-group`

Obligatoire : oui

Assurez-vous de spécifier un groupe de journaux auquel le pilote de journaux `awslogs` envoie ses flux de journaux.

`awslogs-stream-prefix`

Obligatoire : oui

Utilisez l'option `awslogs-stream-prefix` pour associer un flux de journaux au préfixe spécifié, au nom du conteneur et à l'ID de la tâche Amazon ECS à laquelle appartient le conteneur. Si vous spécifiez un préfixe avec cette option, le format du flux de journaux est le suivant.

```
prefix-name/container-name/ecs-task-id
```

Si vous ne spécifiez pas de préfixe avec cette option, le flux de journaux est nommé d'après l'ID de conteneur attribué par le démon Docker sur l'instance de conteneur. Etant

donné qu'il est difficile de suivre les journaux jusqu'au conteneur qui les a envoyé en utilisant uniquement l'ID de conteneur Docker (qui est uniquement disponible sur l'instance de conteneur), nous vous recommandons de spécifier un préfixe avec cette option.

Pour les services Amazon ECS, vous pouvez utiliser le nom du service comme préfixe. Ce faisant, vous pourrez suivre les flux de journaux et déterminer le service auquel appartient le conteneur, trouver le nom du conteneur qui les a envoyés, ainsi que l'ID de la tâche à laquelle le conteneur appartient.

Vous devez spécifier un préfixe de flux pour vos journaux pour qu'ils apparaissent dans le panneau des journaux lors de l'utilisation de la console Amazon ECS.

`awslogs-datetime-format`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes au format `strftime` Python. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Ce format peut, par exemple, servir à analyser une sortie comme une pile de vidage, laquelle pourrait, dans le cas contraire, être consignée en plusieurs entrées. Le modèle adéquat permet de la capturer dans une seule entrée.

Pour plus d'informations, consultez awslogs-datetime-format.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`awslogs-multiline-pattern`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes à l'aide d'une expression régulière. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Pour plus d'informations, consultez awslogs-multiline-pattern.

Cette option est ignorée si `awslogs-datetime-format` est également configuré.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

Les options suivantes s’appliquent à tous les pilotes de journalisation pris en charge.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Cette option définit le mode de transmission des messages de journalisation depuis le conteneur vers le pilote de journalisation spécifié à l’aide de `logDriver`. Le mode de livraison que vous choisissez affecte la disponibilité de l’application lorsque le flux des journaux provenant du conteneur est interrompu.

Si vous utilisez le mode `blocking` et que le flux de journaux est interrompu, les appels provenant du code du conteneur à écrire dans les flux `stdout` et `stderr` seront bloqués. Le fil de journalisation de l'application se bloquera en conséquence. Cela peut empêcher l'application de répondre et entraîner l'échec de vérification de l'état du conteneur.

Si vous utilisez le mode `non-blocking`, les journaux du conteneur sont plutôt stockés dans une mémoire tampon intermédiaire configurée avec l'option `max-buffer-size`. Cela empêche l’application de ne plus répondre lorsque les journaux ne peuvent pas être envoyés. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir

la disponibilité du service et si vous acceptez une certaine perte de journaux. Pour plus d'informations, consultez la section Prévention de la perte de journaux grâce au mode non bloquant dans le pilote de journalisation conteneur `awslogs`.

Vous pouvez définir une valeur de `mode` par défaut pour tous les conteneurs d'un conteneur dans une Région AWS spécifique en utilisant le paramètre de compte `defaultLogDriverMode`. Si vous ne spécifiez pas l'option `mode` dans `logConfiguration` ou ne configurez pas les paramètres du compte, Amazon ECS utilisera par défaut le mode `non-blocking`. Pour plus d'informations sur ce paramètre de compte, consultez la section le Mode de pilote du journal par défaut.

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs, en plus du tampon mémoire du pilote de journalisation.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode par défaut du pilote de journalisation, passant de `blocking` à `non-blocking` afin de privilégier la disponibilité des tâches plutôt que la journalisation. Pour continuer à utiliser le mode `blocking` après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'option `mode` dans votre définition de conteneur `logConfiguration` sur `blocking`.
> - Réglez le paramètre du compte `defaultLogDriverMode` sur `blocking`.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `10m`

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages

intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. Lorsque la mémoire tampon est pleine, il est impossible de stocker d'autres journaux. Les journaux qui ne peuvent pas être stockés sont perdus.

Pour acheminer les journaux à l'aide du routeur de journaux `splunk`, vous devez spécifier un `splunk-token` et une `splunk-url`.

Lorsque vous utilisez le `awsfirelens` routeur de journaux pour acheminer les journaux vers un Service AWS ou AWS Partner Network destination pour le stockage et l'analyse des journaux, vous pouvez définir `log-driver-buffer-limit` option permettant de limiter le nombre de lignes de journal qui sont mises en mémoire tampon avant d'être envoyées au conteneur Log Router. Cela peut aider à résoudre le problème potentiel de perte de journal, car un débit élevé peut entraîner un épuisement de la mémoire pour le tampon à l'intérieur de Docker. Pour plus d'informations, consultez the section called "Configuration des journaux pour un débit élevé".

Les autres options que vous pouvez spécifier lors de l'utilisation d'`awsfirelens` pour acheminer les journaux dépendent de la destination. Lorsque vous exportez des journaux vers Amazon Data Firehose, vous pouvez spécifier Région AWS avec `region` et un nom pour le flux de journaux avec `delivery_stream`.

Lorsque vous exportez des journaux vers Amazon Kinesis Data Streams, vous pouvez spécifier la Région AWS avec `region` et un nom pour le flux de données avec `stream`.

Lorsque vous exportez des journaux vers Amazon OpenSearch Service, vous pouvez spécifier des options telles que `Name`, `Host` (OpenSearch Point de terminaison de service (sans protocole), `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name`, et `tls`.

Lorsque vous exportez des journaux vers Amazon S3, vous pouvez spécifier le compartiment à l'aide de l'option `bucket`. Vous pouvez également spécifier `region`, `total_file_size`, `upload_timeout` et `use_put_object` en tant qu'options.

Ce paramètre nécessite la version 1.19 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`secretOptions`

Type : tableau d'objets

Obligatoire : non

Objet qui représente le secret à transmettre à la configuration de journal. Les secrets utilisés dans la configuration du journal peuvent inclure un jeton d'authentification, un certificat ou une clé de chiffrement. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer à la configuration de journal du conteneur.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tapez : FirelensConfiguration Objet

Obligatoire : non

Le FireLens configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options à utiliser lors de la configuration du log router. Ce champ est facultatif et peut être utilisé pour spécifier un fichier de configuration personnalisé ou ajouter des métadonnées supplémentaires, telles que les détails de tâche, de définition de tâche, de cluster et d'instance de conteneur à l'événement de journal. Si spécifié, la syntaxe à utiliser est `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Pour plus d'informations, consultez Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

`type`

Type : chaîne

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Sécurité

Pour plus d'informations sur la sécurité des conteneurs, consultez la section Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS.

`credentialSpecs`

Type : tableau de chaînes

Obligatoire : non

Une liste d'ARN dans SSM ou Amazon S3 associée à un fichier de spécifications d'informations d'identification (`CredSpec`) qui configure le conteneur pour l'authentification Active Directory. Nous vous recommandons d'utiliser ce paramètre plutôt que les `dockerSecurityOptions`. Le nombre maximal d'ARN est de 1.

Il existe deux formats pour chaque ARN.

credentialspecdomainless:MyARN

Vous utilisez `credentialspecdomainless:MyARN` pour fournir un `CredSpec` avec une section supplémentaire pour un secret dans Secrets Manager. Vous fournissez les informations d'identification de connexion au domaine dans le secret.

Chaque tâche exécutée sur une instance de conteneur peut rejoindre différents domaines.

Vous pouvez utiliser ce format sans associer l'instance de conteneur à un domaine.

credentialspec:MyARN

Vous utilisez `credentialspec:MyARN` pour fournir un `CredSpec` pour un domaine unique.

Vous devez joindre l'instance de conteneur au domaine avant de démarrer toute tâche utilisant cette définition de tâche.

Dans les deux formats, remplacez `MyARN` par l'ARN dans SSM ou Amazon S3.

Le `credspec` doit fournir un ARN dans Secrets Manager pour un secret contenant le nom d'utilisateur, le mot de passe et le domaine auquel se connecter. Pour une meilleure sécurité, l'instance n'est pas jointe au domaine pour l'authentification sans domaine. Les autres applications de l'instance ne peuvent pas utiliser les informations d'identification sans domaine. Vous pouvez utiliser ce paramètre pour exécuter des tâches sur la même instance, même si les tâches doivent être jointes à différents domaines. Pour plus d'informations, veuillez consulter Utilisation de gMSA pour les conteneurs Windows et Utilisation de gMSAs pour les conteneurs Linux.

`user`

Type : chaîne

Obligatoire : non

Utilisateur à utiliser à l'intérieur du conteneur. Ce paramètre correspond à `User` dans la commande docker create-container et à l’option `--user` dans la commande docker run.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0). Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Vous pouvez spécifier l'`user` à l'aide des formats suivants. Si vous spécifiez un UID ou GID, vous devez le spécifier en tant que nombre entier positif.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"user": "string"
```

Limites des ressources

`ulimits`

Type : tableau d'objets

Obligatoire : non

Une liste de valeurs `ulimit` à définir pour un conteneur. Cette valeur remplace le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond à `Ulimits` dans la commande docker create-container et à l'option `--ulimit` dans la commande docker run.

Les tâches Amazon ECS hébergées sur Fargate utilisent les valeurs de limite définies par défaut par le système d'exploitation, à l'exception du paramètre de limite de ressources `nofile`. La limite de ressources `nofile` restreint le nombre de fichiers ouverts qu'un conteneur peut utiliser. Sur Fargate, la limite flexible par défaut `nofile` est `65535` et la limite stricte est `65535`. Vous pouvez définir les valeurs des deux limites jusqu'à `1048576`. Pour plus d'informations, consultez Limites des ressources de tâche.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
   ]
```

`name`

Type : String

Valeurs valides : `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

Le `type` d'`ulimit`.

`hardLimit`

Type : Integer

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite stricte du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en nombre, selon le `type` d'`ulimit`.

`softLimit`

Type : Integer

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite flexible du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en nombre, selon le `type` d'`ulimit`.

Étiquettes Docker

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

UNE key/value carte des étiquettes à ajouter au contenant. Ce paramètre correspond à `Labels` dans la commande docker create-container et à l'option `--label` dans la commande docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

```
"dockerLabels": {"string": "string"
      ...}
```

Autres paramètres de définition de conteneur

Les paramètres de définition de conteneur suivants peuvent être utilisés lors de l'enregistrement des définitions de tâche dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Paramètres Linux
- Dépendances du conteneur
- Temporisations de conteneurs

- Contrôles système
- Interactive
- Pseudo Terminal

Paramètres Linux

`linuxParameters`

Type : objet LinuxParameters

Obligatoire : non

Linux-les options spécifiques appliquées au conteneur, telles que KernelCapabilities.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Type : objet KernelCapabilities

Obligatoire : non

Les fonctionnalités Linux pour le conteneur qui sont supprimées de la configuration par défaut fournie par Docker. Pour plus d'informations sur ces caractéristiques Linux, veuillez consulter la page consacrée aux caractéristiques(7) dans le manuel Linux (langue française non garantie).

`add`

Type : tableau de chaînes

Valeurs valides : `"SYS_PTRACE"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond à `CapAdd` dans la commande docker create-container et à l'option `--cap-add` dans la commande docker run.

`drop`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatoire : non

Caractéristiques Linux du conteneur à supprimer de la configuration par défaut fournie par Docker. Ce paramètre correspond à `CapDrop` dans la commande docker create-container et à l'option `--cap-drop` dans la commande docker run.

`devices`

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond à `Devices` dans la commande docker create-container et `--device` option pour exécuter Docker.

> **Note**
>
> Le paramètre `devices` n'est pas pris en charge lorsque vous utilisez le type de lancement Fargate.

Type : tableau d'objets Périphérique

Obligatoire : non

`hostPath`

Chemin du périphérique dans l'instance de conteneur hôte.

Type : chaîne

Obligatoire : oui

`containerPath`

Chemin auquel exposer l'appareil hôte à l'intérieur du conteneur.

Type : chaîne

Obligatoire : non

`permissions`

Autorisations explicites à fournir au conteneur pour le périphérique. Par défaut, le conteneur dispose des autorisations `read`, `write` et `mknod` sur l'appareil.

Type : tableau de chaînes

Valeurs valides : `read` | `write` | `mknod`

`initProcessEnabled`

Exécutez un processus `init` dans le conteneur afin de transmettre des signaux et de récolter les processus. Ce paramètre correspond à l'option `--init` de docker run.

Ce paramètre nécessite la version 1.25 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`maxSwap`

Il n'est pas pris en charge pour les tâches exécutées sur Fargate.

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est converti en l'option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur et de la valeur `maxSwap`.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur

laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`sharedMemorySize`

Valeur correspondant à la taille (en Mio) du volume `/dev/shm`. Ce paramètre correspond à l'option `--shm-size` de docker run.

> **Note**
>
> Si vous utilisez des tâches qui utilisent Fargate, le paramètre `sharedMemorySize` n'est pas pris en charge.

Type : Integer

`tmpfs`

Chemin du conteneur, options de montage et taille maximale (en Mio) du montage tmpfs. Ce paramètre correspond à l'option `--tmpfs` de docker run.

Type : tableau d'objets Tmpfs

Obligatoire : non

`containerPath`

Chemin de fichier absolu où sera monté le volume tmpfs.

Type : chaîne

Obligatoire : oui

`mountOptions`

Liste des options de montage du volume tmpfs.

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" |`

```
"remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime"
| "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" |
"private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave"
| "relatime" | "norelatime" | "strictatime" | "nostrictatime" |
"mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"
```

`size`

Taille maximale (en Mio) du volume tmpfs.

Type : Integer

Obligatoire : oui

Dépendances du conteneur

`dependsOn`

Type : tableau d'objets ContainerDependency

Obligatoire : non

Dépendances définies pour le démarrage et l'arrêt de conteneurs. Un conteneur peut contenir plusieurs dépendances. Lorsqu'une dépendance est définie pour le démarrage des conteneurs, elle est inversée pour leur arrêt. Pour obtenir un exemple, consultez Dépendances du conteneur.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version ultérieure (Linux) ou `1.0.0` (Windows).

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
```

```
]
```

`containerName`

Type : chaîne

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : chaîne

Obligatoire : oui

Condition de dépendance du conteneur. Voici les conditions disponibles et leur comportement :

- `START` - Cette condition émule le comportement de liens et de volumes aujourd'hui. La condition vérifie qu'un conteneur dépendant est démarré avant d'autoriser d'autres conteneurs à démarrer.
- `COMPLETE` - Cette condition vérifie qu'un conteneur dépendant exécute un cycle complet (sorties) avant d'autoriser d'autres conteneurs à démarrer. Cela peut être utile pour les conteneurs non essentiels qui exécutent un script, puis se ferment. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `SUCCESS` - Cette condition est identique à `COMPLETE`, mais elle nécessite également que le conteneur se termine par un état `zero`. Cette condition ne peut pas être définie sur un contenant essentiel.
- `HEALTHY` - Cette condition vérifie que la surveillance de l'état du conteneur dépendant réussit avant d'autoriser d'autres conteneurs à démarrer. Cela nécessite que le conteneur dépendant dispose de surveillances de l'état configurées dans la définition de la tâche. Cette condition n'est confirmée qu'au démarrage de la tâche.

Temporisations de conteneurs

`startTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée d'attente (en secondes) avant l'abandon de la résolution de dépendances pour un conteneur.

Par exemple, vous spécifiez deux conteneurs dans une définition de tâche, où le `containerA` dispose d'une dépendance au `containerB` avec un état `COMPLETE`, `SUCCESS` ou `HEALTHY`. Si une valeur `startTimeout` est spécifiée pour `containerB` et qu'il n'a pas atteint l'état souhaité dans ce délai, alors le `containerA` ne démarre pas.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version ultérieure (Linux). La valeur maximale est de 120 secondes.

`stopTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée (en secondes) à attendre avant que le conteneur soit expressément tué s'il ne s'achève pas normalement tout seul.

Ce paramètre exige que la tâche ou le service utilise la version de plateforme `1.3.0` ou une version ultérieure (Linux). Si le paramètre n'est pas spécifié, alors la valeur par défaut de 30 secondes est utilisée. La valeur maximale est de 120 secondes.

Contrôles système

`systemControls`

Type : objet SystemControl

Obligatoire : non

Liste des paramètres du noyau de l'espace de noms à définir dans le conteneur. Ce paramètre correspond à `Sysctls` dans la commande docker create-container et à l'option `--sysctl` dans la commande docker run. Par exemple, vous pouvez configurer le paramètre `net.ipv4.tcp_keepalive_time` pour maintenir des connexions de plus longue durée.

Il n'est pas recommandé de spécifier des paramètres `systemControls` liés au réseau pour plusieurs conteneurs dans une seule tâche qui utilise également le mode réseau `awsvpc` ou `host`. En procédant ainsi, vous vous exposez aux inconvénients suivants :

- Si vous définissez `systemControls` pour n'importe quel conteneur, cela s'applique à tous les conteneurs de la tâche. Si vous définissez différents paramètres `systemControls` pour plusieurs conteneurs dans une seule tâche, c'est le conteneur qui est démarré en dernier qui détermine le paramètre `systemControls` qui prend effet.

Si vous définissez un espace de noms des ressources IPC à utiliser pour les conteneurs de la tâche, les conditions suivantes s'appliquent aux contrôles de système. Pour plus d'informations, consultez Mode IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les valeurs des `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

> **Note**
>
> Ce paramètre est uniquement pris en charge pour les tâches hébergées sur AWS Fargate , si elles utilisent la version de plateforme `1.4.0` ou ultérieure (Linux). Cela n'est pas pris en charge par les conteneurs Windows sur Fargate.

```
"systemControls": [
    {
```

```
        "namespace":"string",
        "value":"string"
    }
]
```

`namespace`

Type : chaîne

Obligatoire : non

Paramètre du noyau de l'espace de noms pour lequel définir une `value`.

Valeurs d'espace de noms IPC valides : `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"`, et `Sysctls` qui commencent par `"fs.mqueue.*"`

Valeurs d'espace de noms de réseau valides : `Sysctls` qui commencent par `"net.*"` Sur Fargate, seuls les espaces de noms `Sysctls` existant dans le conteneur sont acceptés.

Toutes ces valeurs sont prises en charge par Fargate.

`value`

Type : chaîne

Obligatoire : non

Valeur du paramètre du noyau de l'espace de noms spécifié dans `namespace`.

Interactive

`interactive`

Type : booléen

Obligatoire : non

Lorsque ce paramètre est `true`, vous pouvez déployer des applications conteneurisées qui nécessitent l'allocation d'une `stdin` ou d'un `tty`. Ce paramètre correspond à `OpenStdin` dans la commande docker create-container et à l'option `--interactive` dans la commande docker run.

La valeur par défaut est `false`.

Pseudo Terminal

`pseudoTerminal`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, un TTY est alloué. Ce paramètre correspond à `Tty` dans la commande docker create-container et à l'option `--tty` dans la commande docker run.

La valeur par défaut est `false`.

Nom de l'accélérateur Elastic Inference (obsolète)

L'exigence de la ressource d'accélérateur Elastic Inference pour votre définition de tâche.

> **Note**
>
> Amazon Elastic Inference (EI) n'est plus disponible pour les clients.

Les paramètres suivants sont autorisés dans une définition de tâche :

`deviceName`

Type : chaîne

Obligatoire : oui

Nom de l'appareil accélérateur d'inférence élastique pour l'instance. `deviceName` doit également être référencé dans une définition du conteneur. Consultez `Elastic Inference accelerator`.

`deviceType`

Type : chaîne

Obligatoire : oui

L'accélérateur Elastic Inference à utiliser.

Configuration du proxy

`proxyConfiguration`

Type : objet ProxyConfiguration

Obligatoire : non

Détails de configuration pour le proxy App Mesh.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Type : String

Valeur valeurs : `APPMESH`

Obligatoire : non

Type de proxy. La seule valeur prise en charge est `APPMESH`.

`containerName`

Type : chaîne

Obligatoire : oui

Nom du conteneur qui fait office de proxy App Mesh.

`properties`

Type : tableau d'objets KeyValuePair

Obligatoire : non

L'ensemble de paramètres de configuration réseau pour fournir le plug-in Container Network Interface (CNI), spécifié sous la forme de paires clé-valeur.

- `IgnoredUID` – (Obligatoire) ID d'utilisateur (UID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredGID`, ce champ doit être vide.
- `IgnoredGID` – (Obligatoire) ID de groupe (GID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredUID`, ce champ doit être vide.
- `AppPorts` - (Obligatoire) Liste des ports utilisés par l'application. Le trafic réseau vers ces ports est transmis à `ProxyIngressPort` et `ProxyEgressPort`.
- `ProxyIngressPort` – (Obligatoire) Spécifie le port auquel le trafic entrant vers `AppPorts` est dirigé.
- `ProxyEgressPort` – (Obligatoire) Spécifie le port auquel le trafic sortant de `AppPorts` est dirigé.
- `EgressIgnoredPorts` – (Obligatoire) Le trafic sortant accédant à ces ports spécifiés est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.
- `EgressIgnoredIPs` – (Obligatoire) Le trafic sortant accédant à ces adresses IP spécifiées est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.

`name`

Type : chaîne

Obligatoire : non

Nom de la paire clé-valeur.

`value`

Type : chaîne

Obligatoire : non

Valeur de la paire clé-valeur.

Volumes

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier une liste de volumes à transmettre au démon Docker sur une instance de conteneur, à laquelle d'autres conteneurs de la même instance de conteneur pourront accéder.

Les types de volumes de données qui peuvent être utilisés sont les suivants :

- Volumes Amazon EBS : fournit un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Vous pouvez attacher un volume Amazon EBS par tâche Amazon ECS lorsque vous exécutez une tâche autonome ou lorsque vous créez ou mettez à jour un service. Les volumes Amazon EBS sont pris en charge pour les tâches Linux hébergées sur Fargate. Pour plus d'informations, consultez Utilisation des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS : fournit un stockage de fichiers simple, évolutif et permanent à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer du stockage dont elles ont besoin, au moment où elles en ont besoin. Les volumes Amazon EFS sont pris en charge pour les tâches hébergées sur Fargate. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.
- Volumes FSx for Windows File Server : fournit des serveurs de fichiers Microsoft Windows entièrement gérés. Ces serveurs de fichiers sont basés sur un système de fichiers Windows. Lorsque vous utilisez FSx for Windows File Server avec Amazon ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d'informations, consultez Utilisation de FSx for Windows File Server avec Amazon ECS.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Montages liés : un fichier ou un dossier sur la machine hôte est monté dans un conteneur. Les volumes hôtes de montage lié sont pris en charge lorsque vous exécutez des tâches. Pour utiliser des volumes hôte de montage lié, spécifiez un `host` et éventuellement une valeur `sourcePath` dans votre définition de tâche.

Pour plus d'informations, consultez Options de stockage pour les tâches Amazon ECS.

Les paramètres suivants sont autorisés dans une définition de conteneur.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union (`-`) et traits de soulignement (_) sont autorisés. Ce nom est référencé dans le paramètre `sourceVolume` de l'objet `mountPoints` de définition du conteneur.

`host`

Obligatoire : non

Le paramètre `host` est utilisé pour lier le cycle de vie du montage lié à l'instance Amazon EC2 hôte, plutôt qu'à la tâche et à l'endroit où elle est stockée. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` Le paramètre est pris en charge uniquement lors de l'utilisation de tâches hébergées sur des instances Amazon EC2 ou des instances gérées Amazon ECS.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le paramètre `host` est utilisé, spécifiez un paramètre `sourcePath` pour déclarer le chemin d'accès sur l'instance Amazon EC2 hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le paramètre `host` contient un emplacement de fichier `sourcePath`, le volume de données persiste à l'emplacement spécifié sur l'instance Amazon EC2 hôte jusqu'à ce que vous le supprimiez manuellement. Si la valeur `sourcePath` n'existe pas sur l'instance Amazon EC2 hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`configuredAtLaunch`

Type : booléen

Obligatoire : non

Spécifie si un volume est configurable au lancement. Lorsqu'il est défini sur `true`, vous pouvez configurer le volume lors de l'exécution d'une tâche autonome ou lors de la création ou de la mise à jour d'un service. Lorsque cette option est définie sur `true`, vous ne pourrez pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être défini sur `true` pour configurer un volume Amazon EBS à rattacher à une tâche. En définissant `configuredAtLaunch` sur `true` et en reportant la configuration du volume à la phase de lancement, vous pouvez créer des définitions de tâches qui ne sont pas limitées à un type de volume ou à des paramètres de volume spécifiques. Cela rend votre définition de tâche réutilisable dans différents environnements d'exécution. Pour plus d'informations, consultez Volumes Amazon EBS.

`dockerVolumeConfiguration`

Tapez : DockerVolumeConfiguration Objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker sont pris en charge uniquement lors de l'exécution de tâches sur des instances EC2. Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : Boolean

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n’est utilisé que si `scope` a la valeur `shared`. Si le champ `scope` a la valeur `task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker, car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l’aide de l’interface CLI du plug-in Docker, utilisez `docker plugin ls` pour récupérer le nom du pilote à partir de votre instance de conteneur. Si le pilote a été installé à l’aide d’une autre méthode, utilisez la découverte du plug-in Docker pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Mappage des options spécifiques au pilote Docker à transmettre. Ce paramètre correspond à `DriverOpts` dans la section « Créer un volume » de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`efsVolumeConfiguration`

Tapez : EFSVolumeConfiguration Objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans `authorizationConfig`, le paramètre de répertoire racine doit être omis ou défini sur `/`, ce qui imposera le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : Integer

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, la tâche utilisera la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Tapez : EFSAuthorizationConfig Objet

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d'accès est spécifié, la valeur du répertoire racine dans `efsVolumeConfiguration` doit être omise ou définie sur `/`, ce qui imposera le chemin défini sur le point d'accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique s'il faut ou non utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

`FSxWindowsFileServerVolumeConfiguration`

Tapez : FSxWindowsFileServerVolumeConfiguration Objet

Obligatoire : oui

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon FSx for Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers FSx for Windows File Server à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : oui

Répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l'intérieur de l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : chaîne

Obligatoire : oui

Options d'informations d'identification d'autorisation.

Options :

- Amazon Resource Name (ARN) d’un secret AWS Secrets Manager.
- ARN d’un paramètre AWS Systems Manager.

`domain`

Type : chaîne

Obligatoire : oui

Nom de domaine complet hébergé par un répertoire AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou un Active Directory EC2 autohébergé.

Étiquettes

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier des étiquettes de métadonnées qui sont appliquées à la définition de tâche. Les balises vous aident à classer et à organiser votre définition de tâche. Chaque balise est constituée d'une clé et d'une valeur facultative. Vous définissez ces deux éléments. Pour plus d'informations, consultez Balisage des ressources Amazon ECS.

> **Important**
>
> N'ajoutez pas de données d'identification personnelle ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreuses personnes AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Les paramètres suivants sont autorisés dans un objet balise.

`key`

Type : chaîne

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : chaîne

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

Autres paramètres de définition de tâche

Les paramètres de définition de tâche suivants peuvent être utilisés lors de l'enregistrement des définitions de tâches dans la console Amazon ECS à l'aide de l'option Configure via JSON

(Configurer via JSON). Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Stockage éphémère
- Mode IPC
- Mode PID
- Injection de pannes

Stockage éphémère

`ephemeralStorage`

Type : objet EphemeralStorage

Obligatoire : non

Quantité de stockage éphémère (en Go) à allouer pour la tâche. Ce paramètre est utilisé pour étendre la quantité totale de stockage éphémère disponible, au-delà de la quantité par défaut, pour les tâches hébergées sur AWS Fargate. Pour plus d'informations, consultez the section called "Montages liés".

> **Note**
>
> Ce paramètre n'est pris en charge que sur la version de plateforme `1.4.0` ou ultérieure (Linux), ou `1.0.0` ou ultérieure (Windows).

Mode IPC

`ipcMode`

Il n'est pas pris en charge pour les tâches exécutées sur Fargate.

Type : chaîne

Obligatoire : non

Espace de noms de ressource IPC à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host`, `task` ou `none`. Si `host` est spécifié, tous les conteneurs des tâches ayant spécifié

le mode IPC `host` sur la même instance de conteneur partagent les mêmes ressources IPC avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent les mêmes ressources IPC. Si `none` est spécifié, les ressources IPC des conteneurs d'une tâche sont privés et ne partagent rien avec les autres conteneurs d'une tâche ou d'une instance de conteneur. Si aucune valeur n'est spécifiée, le partage de l'espace de noms de ressource IPC dépend du paramètre du démon Docker sur l'instance de conteneur.

Si le mode IPC `host` est utilisé, il existe un risque accru d'exposition à un espace de noms IPC indésirable.

Si vous définissez les paramètres du noyau placés dans un espace de noms à l'aide de `systemControls` pour les conteneurs de la tâche, les éléments suivants s'appliquent à votre espace de noms de ressource IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le type de lancement Fargate.

Mode PID

`pidMode`

Type : String

Valeurs valides : `host` | `task`

Obligatoire : non

Espace de noms de processus à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host` ou `task`. Sur les conteneurs Linux, la seule valeur valide est `task`. Par exemple, la surveillance des sidecars peut avoir besoin de `pidMode` pour accéder à des informations sur d'autres conteneurs exécutés dans le cadre de la même tâche.

Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent le même espace de noms de processus.

Si aucune valeur n'est spécifiée, la valeur par défaut est un espace de noms privé pour chaque conteneur.

> **Note**
>
> Ce paramètre est uniquement pris en charge pour les tâches hébergées sur AWS Fargate , si elles utilisent la version de plateforme `1.4.0` ou ultérieure (Linux). Cela n'est pas pris en charge par les conteneurs Windows sur Fargate.

Injection de pannes

`enableFaultInjection`

Type : Boolean

Valeurs valides : `true` | `false`

Obligatoire : non

Si ce paramètre est défini sur `true`, dans les données utiles d'une tâche, Amazon ECS et Fargate acceptent les requêtes d'injection de pannes provenant des conteneurs de la tâche. Par défaut, ce paramètre est défini sur `false`.

Paramètres de définition de tâche Amazon ECS pour Amazon EC2

Les définitions de tâches sont divisées en plusieurs parties : la famille de tâches, le Gestion des identités et des accès AWS le rôle des tâches (IAM), le mode réseau, les définitions des conteneurs, les volumes, les contraintes de placement des tâches et la capacité. Les définitions de famille et de conteneur sont requises dans une définition de tâche. En revanche, le rôle de tâche, le mode réseau, les volumes, les contraintes de placement des tâches et le type de lancement sont facultatifs.

Vous pouvez utiliser ces paramètres dans un fichier JSON pour configurer votre définition de tâche.

Voici des descriptions plus détaillées de chaque paramètre de définition de tâche pour Amazon EC2.

Family

`family`

Type : chaîne

Obligatoire : oui

Lorsque vous enregistrez une définition de tâche, vous lui attribuez une famille. Cela équivaut, pour plusieurs versions de définition de tâche, à lui attribuer un nom spécifié avec un numéro de révision. La première définition de tâche enregistrée dans une famille donnée reçoit le numéro de révision 1. Toute définition de tâche enregistrée après celle-ci reçoit un numéro de révision ultérieur dans l'ordre séquentiel.

Capacity

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la capacité par rapport à laquelle Amazon ECS doit valider la définition de tâche. Une exception client est renvoyée si la définition de tâche n'est pas conforme aux compatibilités spécifiées.

Le paramètre suivant est autorisé dans une définition de tâche.

`requiresCompatibilities`

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `EC2`

La capacité par rapport à laquelle valider la définition de tâche. Une vérification est alors lancée afin de s’assurer que tous les paramètres utilisés dans la définition de tâche répondent aux exigences d’Amazon EC2.

Rôle de tâche

`taskRoleArn`

Type : chaîne

Obligatoire : non

Lorsque vous enregistrez une définition de tâche, vous pouvez fournir un rôle de tâche pour un rôle IAM qui permet aux conteneurs dans l'autorisation de tâche d'appeler en votre nom les API AWS qui sont spécifiées dans les stratégies qui lui associées. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Lorsque vous lancez Amazon ECS-optimized Les rôles IAM et AMI Windows Server pour les tâches sous Windows nécessitent que `-EnableTaskIAMRole` l'option est définie. Vos conteneurs doivent également exécuter certains codes de configuration afin d'utiliser la fonctionnalité. Pour plus d'informations, consultez Configuration supplémentaire d'instance Windows Amazon EC2.

Rôle d'exécution de tâche

`executionRoleArn`

Type : String

Obligatoire : Conditionnelle

Le nom de ressource Amazon (ARN) du rôle d'exécution de la tâche qui autorise l'agent de conteneur Amazon ECS à effectuer AWS Appels d'API en votre nom.

> **Note**
>
> Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Mode réseau

`networkMode`

Type : chaîne

Obligatoire : non

Mode réseau Docker à utiliser pour les conteneurs de la tâche. Pour les tâches Amazon ECS hébergées sur des instances Linux Amazon EC2, les valeurs valides sont `none`, `bridge`, `awsvpc`

et `host`. Si aucun mode réseau n'est spécifié, le mode réseau par défaut est `bridge`. Pour les tâches Amazon ECS hébergées sur des instances Windows Amazon EC2, les valeurs valides sont `default` et `awsvpc`. Si aucun mode réseau n'est spécifié, le mode réseau `default` est utilisé.

Si le mode réseau est défini sur `none`, les conteneurs de la tâche ne disposent d'aucune connectivité externe et les mappages de ports ne peuvent pas être spécifiés dans la définition du conteneur.

Si le mode réseau est `bridge`, la tâche utilise le réseau virtuel intégré de Docker sous Linux, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Linux utilise le pilote réseau Docker `bridge`.

Si le mode réseau est `host`, la tâche utilise le réseau de l'hôte qui contourne le réseau virtuel intégré de Docker en mappant les ports de conteneur directement à l'ENI de l'instance Amazon EC2 qui héberge la tâche. Les mappages de ports dynamiques ne peuvent pas être utilisés dans ce mode réseau. Dans une définition de tâche qui utilise ce mode, un conteneur doit spécifier un numéro `hostPort` spécifique. Un numéro de port sur un hôte ne peut pas être utilisé par plusieurs tâches. Dans ce mode, vous ne pouvez pas exécuter plusieurs tâches de la même définition de tâche sur une instance Amazon EC2 unique.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, vous n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0) pour une meilleure sécurité. Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Si le mode réseau est `awsvpc`, une interface réseau Elastic est attribuée à la tâche, et vous devez spécifier une configuration `NetworkConfiguration` lorsque vous créez un service ou exécutez une tâche avec la définition de tâche. Pour plus d'informations, consultez Options de mise en réseau des tâches Amazon ECS pour EC2.

Si le mode réseau est `default`, la tâche utilise le réseau virtuel intégré de Docker sous Windows, qui s'exécute à l'intérieur de chaque instance Amazon EC2 hébergeant la tâche. Le réseau virtuel intégré sous Windows utilise le pilote réseau Docker `nat`.

Les modes réseau `host` et `awsvpc` offrent les meilleures performances de mise en réseau pour les conteneurs, car ils utilisent la pile de réseau Amazon EC2. Avec les modes réseau

host et awsvpc, les ports de conteneur exposés sont mappés directement au port hôte correspondant (pour le mode réseau hôte host) ou au port de l'interface réseau Elastic (pour le mode réseau awsvpc). Pour cette raison, vous ne pouvez pas utiliser de mappages de ports hôtes dynamiques.

Le mode réseau autorisé dépend du système d'exploitation de l'instance EC2 sous-jacente. Sous Linux, n'importe quel mode réseau peut être utilisé. S'il s'agit de Windows, les modes default et awsvpc peuvent être utilisés.

Plateforme d'exécution

operatingSystemFamily

Type : String

Obligatoire : Conditionnelle

Par défaut : LINUX

Lorsque vous enregistrez une définition de tâche, vous spécifiez la famille du système d'exploitation.

Les valeurs valides sont LINUX, WINDOWS_SERVER_2025_FULL, WINDOWS_SERVER_2025_CORE, WINDOWS_SERVER_2022_CORE, WINDOWS_SERVER_2022_FULL, WINDOWS_SERVER_2019_FULL, WINDOWS_SERVER_2019_CORE, WINDOWS_SERVER_2016_FULL, WINDOWS_SERVER_2004_CORE et WINDOWS_SERVER_20H2_CORE.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsqu'une définition de tâche fait partie d'un service, cette valeur doit correspondre à la valeur platformFamily du service.

cpuArchitecture

Type : String

Obligatoire : Conditionnelle

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier l'architecture du processeur. Les valeurs valides sont X86_64 et ARM64. Si vous ne spécifiez aucune valeur,

Amazon ECS tente de placer les tâches sur l'architecture de processeur disponible en fonction de la configuration du fournisseur de capacité. Pour vous assurer que les tâches sont placées sur une architecture de processeur spécifique, spécifiez une valeur pour `cpuArchitecture` dans la définition de la tâche.

Toutes les définitions de tâches qui sont utilisées dans un service doivent avoir la même valeur pour ce paramètre.

Lorsque vous avez des tâches Linux, vous pouvez définir la valeur sur `ARM64`. Pour plus d'informations, consultez the section called “Définitions de tâche pour les charges de travail ARM 64 bits”.

Taille de la tâche

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier la quantité totale d'UC et de mémoire utilisée pour cette tâche. Ces valeurs sont distinctes des valeurs `cpu` et `memory` au niveau de la définition de conteneur. Pour les tâches hébergées sur des instances Amazon EC2, ces champs sont facultatifs.

> **Note**
>
> Task-level Les paramètres du processeur et de la mémoire sont ignorés pour les conteneurs Windows. Nous vous recommandons de spécifier des ressources de niveau conteneur pour les conteneurs Windows.

`cpu`

Type : String

Obligatoire : Conditionnelle

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Limite stricte du nombre d'unités UC à présenter pour la tâche. Vous pouvez spécifier les valeurs de l'UC dans le fichier JSON sous forme de chaîne en unités d'UC ou en processeurs virtuels

(vCPU). Par exemple, vous pouvez spécifier une valeur d'UC de `1024` en unités d'UC ou de `1 vCPU` en vCPU. Lorsque la définition de tâche est enregistrée, la valeur de vCPU (processeurs virtuels) est convertie en un nombre entier indiquant les unités UC.

Ce champ est facultatif. Si votre cluster n'a pas d'instances de conteneur enregistrées avec les unités UC demandées disponibles, la tâche échoue. Les valeurs prises en charge se situent entre `0.125` vCPU et `192` vCPU.

`memory`

Type : String

Obligatoire : Conditionnelle

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Limite stricte de quantité de mémoire à allouer à la tâche. Vous pouvez spécifier les valeurs de mémoire dans la définition de la tâche sous forme de chaîne en mébioctets (Mio) ou en gigaoctets (Go). Par exemple, vous pouvez spécifier une valeur de mémoire de `3072` en Mio ou de `3 GB` en Go. Lorsque la définition de tâche est enregistrée, la valeur en Go est convertie en un nombre entier indiquant la quantité de Mio.

Ce champ est facultatif et n'importe quelle valeur peut être utilisée. Si une valeur de mémoire au niveau de la tâche est spécifiée, la valeur de mémoire au niveau du conteneur est facultative. Si votre cluster n'a pas d'instances de conteneur enregistrées avec la mémoire demandée disponible, la tâche échoue. Vous pouvez optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier. Pour plus d'informations, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

Définitions de conteneur

Lorsque vous enregistrez une définition de tâche, vous devez spécifier une liste de définitions de conteneur qui sont transmises au démon Docker sur une instance de conteneur. Les paramètres suivants sont autorisés dans une définition de conteneur.

Rubriques

- Paramètres de définition de conteneur standards
- Paramètres de définition de conteneur avancés
- Autres paramètres de définition de conteneur

Paramètres de définition de conteneur standards

Les paramètres de définition de tâche suivants sont obligatoires ou utilisés dans la plupart des définitions de conteneur.

Rubriques

- Nom
- Image
- Mémoire
- Cartographie des ports
- Informations d'identification du référentiel privé

Nom

`name`

Type : chaîne

Obligatoire : oui

Le nom d'un conteneur. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés. Si vous associez plusieurs conteneurs dans une définition de tâche, vous pouvez spécifier l'option `name` d'un conteneur dans l'option `links` d'un autre conteneur. Cela permet de connecter les conteneurs.

Image

`image`

Type : chaîne

Obligatoire : oui

Image utilisée pour démarrer un conteneur. Cette chaîne est transmise directement au démon Docker. Par défaut, les images dans le registre Docker Hub sont disponibles. Vous pouvez également spécifier d'autres référentiels avec soit *repository-url*/*image*:*tag*, soit *repository-url*/*image*@*digest*. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, deux points, des points, des barres obliques et des signes dièse. Ce paramètre correspond à `Image` dans la commande docker create-container et `IMAGE` paramètre de la commande Docker run.

- Lorsqu'une nouvelle tâche démarre, l'agent de conteneur Amazon ECS extrait la version la plus récente de l'image et de l'étiquette spécifiées afin que le conteneur puisse les utiliser. Notez cependant que les mises à jour ultérieures apportées à une image de référentiel ne sont pas répercutées sur les tâches déjà en cours d'exécution.
- Lorsque vous ne spécifiez pas de balise ou de résumé dans le chemin de l'image dans la définition de la tâche, l'agent de conteneur Amazon ECS utilise le `latest` balise pour extraire l'image spécifiée.
- Les mises à jour ultérieures apportées à une image de référentiel ne sont pas propagées aux tâches déjà en cours d'exécution.
- Les images des registres privés sont prises en charge. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.
- Les images des référentiels Amazon ECR peuvent être spécifiées en utilisant soit la convention de dénomination complète `registry/repository:tag` ou `registry/repository@digest` (par exemple, *aws_account_id*`.dkr.ecr.`*region*`.amazonaws.com/`*my-web-app*`:`*latest* ou *aws_account_id*`.dkr.ecr.`*region*`.amazonaws.com/`*my-web-app*`@`*sha256:94afd1f2e64d908bc90dbca0035a5b567EXAMPLE*).
- Les images dans les référentiels officiels sur Docker Hub utilisent un nom unique (par exemple, `ubuntu` ou `mongo`).
- Les images dans les autres référentiels sur Docker Hub sont qualifiées par un nom d'organisation (par exemple, `amazon/amazon-ecs-agent`).
- Les images dans les autres référentiels en ligne sont qualifiées par un nom de domaine (par exemple, `quay.io/assemblyline/ubuntu`).

`versionConsistency`

Type : String

Valeurs valides : `enabled`|`disabled`

Obligatoire : non

Spécifie si Amazon ECS résoudra la balise d'image de conteneur fournie dans la définition du conteneur en un condensé d'image. Par défaut, ce comportement est `enabled`. Si vous définissez la valeur d'un conteneur sur `disabled`, Amazon ECS ne résoudra pas la balise d'image de conteneur en un condensé et utilisera l'URI d'image d'origine spécifiée dans la définition du conteneur pour le déploiement. Pour plus d'informations sur la résolution des images de conteneur, consultez la section Résolution des images de conteneurs.

Mémoire

`memory`

Type : Integer

Obligatoire : non

La quantité de mémoire (en Mio) à présenter au conteneur. Si votre conteneur tente de dépasser la mémoire spécifiée ici, il sera désactivé. La quantité totale de mémoire réservée pour tous les conteneurs au sein d'une tâche doit être inférieure à la valeur `memory` de la tâche, si cette valeur est spécifiée. Ce paramètre correspond à `Memory` dans la commande docker create-container et à l'option `--memory` dans la commande docker run.

Vous devez spécifier une valeur de mémoire soit au niveau de la tâche, soit au niveau du conteneur. Si vous spécifiez à la fois une valeur de `memory` au niveau du conteneur et une valeur `memoryReservation`, la valeur de `memory` doit être supérieure à celle de `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

`memoryReservation`

Type : Integer

Obligatoire : non

La limite flexible (en Mio) de mémoire à réserver pour le conteneur. En cas de contention de la mémoire système, Docker tente de garder la mémoire du conteneur en deçà de cette limite flexible. Toutefois, votre conteneur peut utiliser davantage de mémoire en cas de besoin. Le conteneur peut consommer jusqu'à la limite stricte spécifiée avec le paramètre `memory` (le cas échéant), ou la totalité de la mémoire disponible sur l'instance de conteneur, selon la première valeur atteinte. Ce paramètre correspond à `MemoryReservation` dans la commande docker create-container et à l'option `--memory-reservation` dans la commande docker run.

Si aucune valeur de mémoire au niveau de la tâche n'est spécifiée, vous devez indiquer un nombre entier différent de zéro pour `memory` ou `memoryReservation` dans une définition de conteneur. Si vous spécifiez les deux, `memory` doit être supérieur à `memoryReservation`. Si vous spécifiez `memoryReservation`, cette valeur est soustraite des ressources mémoire disponibles pour l'instance de conteneur sur laquelle le conteneur est placé. Sinon, c'est la valeur `memory` qui est utilisée.

Par exemple, supposons que votre conteneur utilise normalement 128 Mio de mémoire, mais qu'il lui arrive d'utiliser jusqu'à 256 Mio de mémoire pendant de courtes périodes. Vous pouvez définir une `memoryReservation` de 128 Mio et une limite stricte `memory` de 300 Mio. Cette configuration permet au conteneur de ne réserver que 128 Mio de mémoire à partir des ressources restantes sur l'instance de conteneur. En même temps, cette configuration permet également au conteneur d'utiliser davantage de ressources mémoire lorsque cela est nécessaire.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Le démon Docker 20.10.0 ou ultérieur réserve un minimum de 6 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 6 Mio de mémoire pour vos conteneurs.

Le démon Docker 19.03.13-ce ou antérieur réserve un minimum de 4 Mio de mémoire pour un conteneur. Par conséquent, ne spécifiez pas moins de 4 Mio de mémoire pour vos conteneurs.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

Cartographie des ports

`portMappings`

Type : tableau d'objets

Obligatoire : non

Les mappages de ports exposent les ports réseau de votre conteneur au monde extérieur, ce qui permet aux clients d'accéder à votre application. Ils sont également utilisés pour la communication entre conteneurs au sein d'une même tâche.

Pour les définitions de tâche qui utilisent le mode réseau `awsvpc`, spécifiez uniquement `containerPort`. Le `hostPort` est toujours ignoré et le port du conteneur est automatiquement mappé à un port aléatoire portant un numéro élevé sur l'hôte.

Les mappages de ports sur Windows utilisent l'adresse de passerelle `NetNAT` plutôt que `localhost`. Il n'existe aucune boucle pour les mappages de ports sous Windows, donc vous ne pouvez pas accéder au port mappé d'un conteneur à partir de l'hôte lui-même.

La plupart des champs de ce paramètre (y compris `containerPort`, `hostPort`, `protocol`) carte vers `PortBindings` dans la commande docker create-container et `--publish` option pour exécuter Docker. Si le mode réseau d'une définition de tâche est défini sur `host`, les ports hôtes doivent être indéfinis ou correspondre au port du conteneur dans le mappage de port.

> **Note**
>
> Une fois qu'une tâche passe à l'état `RUNNING`, les affectations manuelles et automatiques de ports de conteneur et d'hôte sont visibles aux emplacements suivants :
>
> - Console : la section Network Bindings (Liaisons réseau) de la description d'un conteneur pour une tâche sélectionnée.
> - AWS CLI : la section `networkBindings` de la sortie de la commande describe-tasks.
> - API : la réponse `DescribeTasks`.
> - Métadonnées : point de terminaison des métadonnées de la tâche.

`appProtocol`

Type : chaîne

Obligatoire : non

Protocole d'application utilisé pour le mappage de port. Ce paramètre s'applique uniquement à Service Connect. Nous vous conseillons de définir ce paramètre de manière cohérente avec le protocole que votre application utilise. Si vous définissez ce paramètre, Amazon ECS ajoute une gestion de connexion spécifique au protocole au proxy Service Connect. Si vous définissez ce paramètre, Amazon ECS ajoute une télémétrie spécifique au protocole dans la console Amazon ECS et CloudWatch.

Si vous ne définissez aucune valeur pour ce paramètre, le protocole TCP est utilisé. Toutefois, Amazon ECS n'ajoute pas de télémétrie spécifique au protocole pour TCP.

Pour plus d'informations, consultez the section called “ Connexion au service”.

Valeurs de protocole valides : `"http" | "http2" | "grpc"`

`containerPort`

Type : Integer

Obligatoire : oui, lorsque des objets `portMappings` sont utilisés

Le numéro de port sur le conteneur qui est lié au port hôte spécifié par l'utilisateur ou affecté automatiquement.

Pour les tâches utilisant le mode réseau `awsvpc`, vous utilisez `containerPort` pour spécifier les ports exposés.

Supposons que vous utilisiez des conteneurs dans une tâche avec les fournisseurs de capacité EC2 et que vous spécifiez un port de conteneur et non un port hôte. Ensuite, votre conteneur reçoit automatiquement un port hôte dans la plage de ports éphémères. Pour plus d'informations, consultez `hostPort`. Les mappages de ports qui sont automatiquement affectés de cette façon ne sont pas comptabilisés dans le quota des 100 ports réservés d'une instance de conteneur.

`containerPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement.

Vous ne pouvez définir ce paramètre qu'à l'aide de l'API `register-task-definition`. L'option est disponible dans le paramètre `portMappings`. Pour plus d'informations, consultez register-task-definition dans la Référence AWS Command Line Interface .

Les règles suivantes s'appliquent lorsque vous spécifiez une `containerPortRange` :

- Vous devez utiliser le mode réseau `bridge` ou le mode réseau `awsvpc`.
- Ce paramètre est disponible pour les systèmes d'exploitation Windows et Linux.
- L'instance de conteneur doit au moins disposer de la version 1.67.0 de l'agent de conteneur et au moins de la version 1.67.0-1 du package `ecs-init`.
- Vous pouvez spécifier un maximum de 100 plages de ports pour chaque conteneur.
- Vous ne spécifiez pas de `hostPortRange`. La valeur de `hostPortRange` est définie comme suit :
  - Pour les conteneurs d'une tâche en mode réseau `awsvpc`, le `hostPort` est défini sur la même valeur que le `containerPort`. Il s'agit d'une stratégie de mappage statique.
  - Pour les conteneurs dans le cadre d'une tâche avec le `bridge` en mode réseau, l'agent Amazon ECS trouve les ports hôtes ouverts dans la plage éphémère par défaut et les transmet à Docker pour les lier aux ports du conteneur.
- Les valeurs valides de `containerPortRange` sont comprises entre 1 et 65535.
- Un port peut uniquement être inclus dans un mappage de port pour chaque conteneur.

- Vous ne pouvez pas spécifier de plages de ports qui se chevauchent.
- Le premier port de la plage doit être inférieur au dernier port de la plage.
- Docker vous recommande de désactiver le Docker-proxy dans le Docker fichier de configuration du démon lorsque vous avez un grand nombre de ports.

  Pour plus d'informations, voir Numéro #11185 sur GitHub.

  Pour plus d'informations sur la désactivation du proxy Docker dans le fichier de configuration du démon Docker, veuillez consulter Démon Docker dans le Guide du développeur Amazon ECS (langue française non garantie).

Vous pouvez appeler DescribeTasks pour voir la `hostPortRange` correspondant aux ports hôtes liés aux ports de conteneur.

Les plages de ports ne sont pas incluses dans les événements de tâches Amazon ECS, qui sont envoyés à EventBridge. Pour plus d'informations, consultez the section called "Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge".

`hostPortRange`

Type : chaîne

Obligatoire : non

Plage de numéros de port sur l'hôte utilisée avec la liaison réseau. Elle est attribuée par Docker et délivrée par l'agent Amazon ECS.

`hostPort`

Type : Integer

Obligatoire : non

Le numéro de port sur l'instance de conteneur à réserver pour votre conteneur.

Vous pouvez spécifier un port hôte non réservé pour le mappage de ports de votre conteneur. C'est ce que l'on appelle le mappage statique des ports hôtes. Vous pouvez également omettre le `hostPort` (ou le définir sur `0`) lorsque vous spécifiez un `containerPort`. Votre conteneur reçoit automatiquement un port dans la plage de ports éphémères pour le système d'exploitation et la version Docker de votre instance de conteneur. C'est ce que l'on appelle le mappage dynamique des ports hôtes.

La plage de ports éphémères par défaut pour Docker 1.6.0 et versions ultérieures est répertoriée dans l'instance sous `/proc/sys/net/ipv4/ip_local_port_range`. Si ce paramètre de noyau n'est pas disponible, la plage de ports éphémères par défaut de `49153–65535` est utilisée. N'essayez pas de spécifier un port d'hôte dans la plage de ports éphémères. Cela est dû au fait qu'ils sont réservés à une affectation automatique. En général, les ports inférieurs à `32768` ne sont pas compris dans la plage de ports éphémères. Vous pouvez utiliser `ECS_DYNAMIC_HOST_PORT_RANGE` paramètre dans la configuration de l'agent de conteneur ECS pour spécifier une plage personnalisée pour les ports hôtes attribués dynamiquement. Cela peut être utile si vos tâches ne démarrent pas en raison de conflits de ports avec d'autres processus de l'instance de conteneur, tels que des connexions sortantes qui utilisent également des ports de la plage de ports éphémères. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Les ports réservés par défaut sont le port `22` pour SSH, les ports Docker `2375` et `2376`, et les ports d'agent de conteneur Amazon ECS `51678-51680`. Tout port hôte ayant été spécifié précédemment par l'utilisateur pour une tâche en cours d'exécution est également réservé pendant l'exécution de la tâche. Après l'arrêt d'une tâche, le port hôte est libéré. Les ports réservés actuels s'affichent dans les `remainingResources` de la sortie de describe-container-instances. Une instance de conteneur peut avoir jusqu'à 100 ports réservés à la fois, y compris les ports réservés par défaut. Les ports affectés automatiquement ne sont pas pris en compte dans le quota des 100 ports réservés.

`name`

Type : String

Obligatoire : Non, requis pour que Service Connect et VPC Lattice soient configurés dans un service

Nom utilisé pour le mappage de port. Ce paramètre ne s'applique qu'à Service Connect et VPC Lattice. Ce paramètre correspond au nom que vous utilisez dans la configuration Service Connect et VPC Lattice d'un service.

Pour plus d'informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.

Dans l'exemple suivant, les deux champs obligatoires pour Service Connect et VPC Lattice sont utilisés.

```
"portMappings": [
    {
        "name": string,
        "containerPort": integer
    }
]
```

`protocol`

Type : chaîne

Obligatoire : non

Le protocole utilisé pour le mappage de port. Les valeurs valides sont `tcp` et `udp`. La valeur par défaut est `tcp`.

> **Important**
>
> Seul `tcp` est pris en charge pour Service Connect. N'oubliez pas que `tcp` est implicite si ce champ n'est pas défini.

> **Important**
>
> La prise en charge du protocole UDP n'est disponible que sur les instances de conteneur qui ont été lancées avec la version 1.2.0 ou ultérieure de l'agent de conteneur Amazon ECS (comme l'AMI `amzn-ami-2015.03.c-amazon-ecs-optimized`), ou avec des agents de conteneur qui ont été mis à jour vers la version 1.3.0 ou ultérieure. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Si vous spécifiez un port hôte, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer,
        "hostPort": integer
    }
    ...
```

```
]
```

Si vous souhaitez utiliser un port hôte affecté automatiquement, utilisez la syntaxe suivante.

```
"portMappings": [
    {
        "containerPort": integer
    }
    ...
]
```

Informations d'identification du référentiel privé

`repositoryCredentials`

Type : objet RepositoryCredentials

Obligatoire : non

Informations d'identification du référentiel pour l'authentification de registre privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

`credentialsParameter`

Type : String

Obligatoire : oui, lorsque des objets `repositoryCredentials` sont utilisés

L'Amazon Resource Name (ARN) du secret contenant les informations d'identification du référentiel privé.

Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

> **Note**
>
> Lorsque vous utilisez l'API Amazon ECS, AWS CLI, ou AWS SDK, si le secret existe dans la même région que la tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Lorsque vous utilisez Console de gestion AWS, vous devez spécifier l'ARN complet du secret.

Voici un extrait de code d'une définition de tâche indiquant les paramètres requis :

```
"containerDefinitions": [
    {
        "image": "private-repo/private-image",
        "repositoryCredentials": {
            "credentialsParameter":
 "arn:aws:secretsmanager:region:aws_account_id:secret:secret_name"
        }
    }
]
```

Paramètres de définition de conteneur avancés

Les paramètres avancés de définition de conteneur suivants fournissent des fonctionnalités étendues à la commande Docker run qui est utilisée pour lancer des conteneurs sur vos instances de conteneurs Amazon ECS.

Rubriques

- Politique de redémarrage
- Surveillance de l'état
- Environnement
- Paramètres réseau
- Stockage et journalisation
- Sécurité
- Limites des ressources
- Étiquettes Docker

Politique de redémarrage

`restartPolicy`

La politique de redémarrage du conteneur et les paramètres de configuration associés. Lorsque vous configurez une politique de redémarrage pour un conteneur, Amazon ECS peut redémarrer le conteneur sans avoir à remplacer la tâche. Pour plus d'informations, consultez Redémarrage

de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`enabled`

Type : Boolean

Obligatoire : oui

Spécifie si une politique de redémarrage est activée pour le conteneur.

`ignoredExitCodes`

Type : tableau integer

Obligatoire : non

Liste des codes de sortie qu'Amazon ECS ignorera et pour lesquels il ne tentera pas de redémarrage. Vous pouvez spécifier un maximum de 50 codes de sortie de conteneur. Par défaut, Amazon ECS n'ignore aucun code de sortie.

`restartAttemptPeriod`

Type : Integer

Obligatoire : non

Durée (en secondes) pendant laquelle le conteneur doit fonctionner avant de pouvoir tenter un redémarrage. Un conteneur ne peut être redémarré qu'une fois toutes les `restartAttemptPeriod` secondes. Si un conteneur n'est pas en mesure de fonctionner pendant cette période et qu'il se ferme prématurément, il ne sera pas redémarré.
Vous pouvez définir une `restartAttemptPeriod` minimale de 60 secondes et une `restartAttemptPeriod` maximale de 1 800 secondes. Par défaut, un conteneur doit fonctionner pendant 300 secondes avant de pouvoir être redémarré.

Surveillance de l'état

`healthCheck`

Commande de surveillance de l'état du conteneur et paramètres de configuration associés pour le conteneur. Pour plus d'informations, consultez Détermination de l'état des tâches Amazon ECS à l'aide de la surveillance de l'état des conteneurs.

`command`

Tableau de chaînes représentant la commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur. Si vous n'en spécifiez aucun, `CMD` est utilisé.

Lors de l'enregistrement d'une définition de tâche dans le Console de gestion AWS, utilisez une liste de commandes séparées par des virgules. Ces commandes sont converties en une chaîne une fois la définition de tâche créée. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
CMD-SHELL, curl -f http://localhost/ || exit 1
```

Lors de l'enregistrement d'une définition de tâche à l'aide du Console de gestion AWS panneau JSON, le AWS CLI, ou les API, placez la liste des commandes entre crochets. Un exemple d'entrée pour une surveillance de l'état est le suivant.

```
[ "CMD-SHELL", "curl -f http://localhost/ || exit 1" ]
```

Un code de sortie de 0, avec aucune sortie `stderr`, indique la réussite et un code de sortie autre que zéro indique qu'il s'agit d'un échec.

`interval`

La durée (en secondes) entre chaque surveillance de l'état. Vous pouvez indiquer une durée comprise entre 5 et 300 secondes. La valeur par défaut est de 30 secondes.

`timeout`

La durée (en secondes) d'attente pour qu'une surveillance de l'état réussisse avant qu'elle ne soit considérée comme un échec. Vous pouvez indiquer une durée comprise entre 2 et 60 secondes. La valeur par défaut est de 5 secondes.

`retries`

Nombre de nouvelles tentatives d'exécution d'une surveillance de l'état ayant échoué avant que le conteneur soit considéré comme défectueux. Vous pouvez indiquer un nombre de tentatives compris en 1 et 10. La valeur par défaut est de trois tentatives.

`startPeriod`

La période de grâce facultative pour donner aux conteneurs le temps de démarrer avant l'échec des surveillance de l'état est prise en compte dans le nombre maximal de tentatives. Vous pouvez indiquer une durée comprise entre 0 et 300 secondes. Par défaut, la `startPeriod` est désactivée.

Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Environnement

`cpu`

Type : Integer

Obligatoire : non

Le nombre d'unités `cpu` que l'agent de conteneur Amazon ECS réserve pour le conteneur. Sous Linux, ce paramètre correspond à `CpuShares` dans la section Créer un conteneur.

> **Note**
>
> Vous pouvez déterminer le nombre d'unités de processeur qui sont disponibles pour chaque type d'instance Amazon EC2. Pour ce faire, multipliez par 1 024 le nombre de vCPU répertoriés pour ce type d'instance sur la page détaillée Instances Amazon EC2.

Les conteneurs Linux partagent des unités de processeur non allouées avec les autres conteneurs de l'instance de conteneur selon le même ratio que la quantité allouée. Par exemple, supposons que vous exécutez une tâche à conteneur unique sur un type d'instance à cœur unique avec 512 unités de processeur spécifiées pour ce conteneur. De plus, cette tâche est la seule tâche en cours d'exécution sur l'instance de conteneur. Dans cet exemple, le conteneur peut utiliser le partage complet de 1 024 unités UC à tout moment. Toutefois, supposons que vous ayez lancé une autre copie de la même tâche sur cette instance de conteneur. Un minimum de 512 unités de processeur est affecté à chaque tâche si nécessaire. De même, si l'autre conteneur n'utilise pas le processeur restant, chaque conteneur peut bénéficier d'une utilisation plus

importante du processeur. Toutefois, si les deux tâches sont actives à 100 % en permanence, elles sont limitées à 512 unités d'UC.

Sur les instances de conteneur Linux, le démon Docker de l'instance de conteneur se sert de la valeur de processeur pour calculer les ratios de partage de processeur relatifs pour les conteneurs en cours d'exécution. La valeur de parts d'UC minimale autorisée par le noyau Linux est de 2, et la valeur de parts d'UC maximale autorisée par le noyau Linux est de 262 144. Toutefois, le paramètre UC n'est pas obligatoire et vous pouvez utiliser des valeurs d'UC inférieures à 2 ou supérieures à 262 144 dans vos définitions de conteneur. Pour les valeurs d'UC inférieures à 2 (y compris null) ou supérieures à 262 144, le comportement varie selon la version de l'agent de conteneur Amazon ECS :

- Versions de l'agent <= 1.1.0 : les valeurs de processeur nulles et égales à zéro sont transmises à Docker en tant que 0, ce que Docker convertit ensuite en 1 024 parts de processeur. Les valeurs de processeur de 1 sont transmises à Docker en tant que 1, ce que le noyau Linux convertit en deux parts de processeur.
- Versions de l'agent > = 1.2.0 : les valeurs de processeur nulles, 0 et 1 sont transmises à Docker en tant que deux partages de processeur..
- Versions d'agent >= 1.84.0 : les valeurs d'UC supérieures à 256 vCPU sont transmises à Docker avec la valeur de 256, ce qui équivaut à 262 144 partages d'UC.

Sur les instances de conteneur Windows, le quota de processeur est appliqué comme quota absolu. Les conteneurs Windows n'ont accès qu'à la quantité spécifiée de processeur décrite dans la définition de tâche. Une valeur de processeur nulle ou égale à zéro est transmise à Docker comme étant `0`. Windows interprète ensuite cette valeur comme 1 % d'un processeur.

Pour voir d'autres exemples, veuillez consulter le billet de blog How Amazon ECS manages CPU and memory resources.

`gpu`

Type : objet ResourceRequirement

Obligatoire : non

Le nombre de `GPUs` physiques que l'agent de conteneur Amazon ECS réserve pour le conteneur. Vous pouvez spécifier une valeur entière, une valeur décimale pour les GPU fractionnaires ou `ALL`. Lorsque vous spécifiez `ALL`, tous les GPU de l'instance de conteneur sont alloués au conteneur. Lorsque vous spécifiez une valeur décimale (telle que `0.125`, `0.25`, ou `0.5`), Amazon

ECS place la tâche sur une instance GPU fractionnée (G6f) qui fournit la capacité GPU requise. Le nombre de GPU réservés pour tous les conteneurs dans une tâche ne doit pas dépasser le nombre de GPU disponibles sur l'instance de conteneur sur laquelle la tâche est lancée. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

`sharingStrategy`

Type : objet SharingStrategy

Obligatoire : non

Configuration de partage du GPU pour le conteneur. Spécifiez `sharingStrategy` pour partager un seul GPU physique sur plusieurs conteneurs avec NVIDIA Multi-Process Service (MPS). Cet objet contient `nvidiaMps` objet avec les champs suivants :

Sur un GPU besoins en ressources, `value` et `sharingStrategy` s'excluent mutuellement. Lorsque vous spécifiez `sharingStrategy`, vous devez omettre `value`.

`memory`

Type : Integer

Obligatoire : oui

Limite de mémoire GPU pour le conteneur, en MiB. Cette valeur représente un plafond strict de la mémoire GPU que le conteneur peut utiliser. La valeur doit être supérieure à `0`.

`maxComputePercent`

Type : Integer

Obligatoire : non

Le pourcentage maximum de calcul GPU que le conteneur peut utiliser, sous la forme d'un entier de 1 pour `100`. Cette valeur est un plafond et non une réservation. Si vous l'omettez, le conteneur peut utiliser jusqu'à `100` pour cent.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

`Elastic Inference accelerator`

Type : objet ResourceRequirement

Obligatoire : non

Pour le type `InferenceAccelerator`, `value` correspond à `deviceName` d'un `InferenceAccelerator` spécifié dans une définition de tâche. Pour plus d'informations, consultez the section called "Nom de l'accélérateur Elastic Inference (obsolète)".

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

`essential`

Type : booléen

Obligatoire : non

Supposons que le paramètre `essential` d'un conteneur soit marqué comme `true`, et que ce conteneur échoue ou s'arrête pour une raison quelconque. Ensuite, tous les autres conteneurs qui font partie de la tâche sont arrêtés. Si le paramètre `essential` d'un conteneur a la valeur `false`, son échec n'a pas d'incidence sur le reste des conteneurs dans la tâche. Si ce paramètre n'est pas spécifié, le conteneur est supposé essentiel.

Toutes les tâches doivent comporter au moins un conteneur essentiel. Supposons que vous disposiez d'une application composée de plusieurs conteneurs. Ensuite, les conteneurs de groupes qui sont utilisés dans un même but pour former des composants, et séparer les différents composants en plusieurs définitions de tâches. Pour plus d'informations, consultez Architecture de votre application pour Amazon ECS.

```
"essential": true|false
```

`entryPoint`

> **Important**
>
> Les premières versions de l'agent de conteneur Amazon ECS ne traitent pas correctement les paramètres `entryPoint`. Si vous rencontrez des difficultés pour utiliser `entryPoint`, mettez à jour l'agent de conteneur ou saisissez plutôt vos commandes et vos arguments sous forme d'éléments de tableau `command`.

Type : tableau de chaînes

Obligatoire : non

Point d'entrée qui est transmis au conteneur.

```
"entryPoint": ["string", ...]
```

`command`

Type : tableau de chaînes

Obligatoire : non

La commande transmise au conteneur. Ce paramètre correspond à `Cmd` dans la commande create-container et au paramètre `COMMAND` de docker run. S'il existe plusieurs arguments, assurez-vous que chaque argument est une chaîne séparée dans le tableau.

```
"command": ["string", ...]
```

`workingDirectory`

Type : chaîne

Obligatoire : non

Répertoire de travail dans lequel exécuter les commandes à l'intérieur du conteneur. Ce paramètre correspond à `WorkingDir` dans la section Create a container (Créer un conteneur) de l'API Docker à distance et l'option `--workdir` correspond à docker run.

```
"workingDirectory": "string"
```

`environmentFiles`

Type : tableau d'objets

Obligatoire : non

Liste des fichiers contenant les variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `--env-file` option de la commande Docker run.

Lorsque le protocole FIPS est activé, les noms de compartiments comportant des points (.) (par exemple, amzn-s3-demo-bucket1.name.example) ne sont pas pris en charge. La présence de points (.) dans le nom du compartiment empêche le démarrage de la tâche, car l'agent ne peut pas extraire le fichier de variables d'environnement depuis Amazon S3.

Ceci n'est pas disponible pour les conteneurs Windows.

Vous pouvez spécifier jusqu'à dix fichiers d'environnement. Le fichier doit avoir une extension de fichier `.env`. Chaque ligne d'un fichier d'environnement contient une variable d'environnement au format `VARIABLE=VALUE`. Les lignes commençant par # sont traitées comme des commentaires et sont ignorées.

Si des variables d'environnement individuelles sont spécifiées dans la définition du conteneur, elles ont priorité sur les variables contenues dans un fichier d'environnement. Si plusieurs fichiers d'environnement contenant la même variable sont spécifiés, ils sont traités de haut en bas. Nous vous recommandons d'utiliser des noms de variables uniques. Pour plus d'informations, consultez Transmission d'une variable d'environnement individuelle à un conteneur Amazon ECS.

`value`

Type : chaîne

Obligatoire : oui

L'Amazon Resource Name (ARN) de l'objet Amazon S3 contenant le fichier de variable d'environnement.

`type`

Type : chaîne

Obligatoire : oui

Type de fichier à utiliser La seule valeur prise en charge est `s3`.

`environment`

Type : tableau d'objets

Obligatoire : non

Variables d'environnement à transmettre à un conteneur. Ce paramètre correspond à `Env` dans la commande docker create-container et à l'option `--env` dans la commande docker run.

> **Important**
>
> Nous déconseillons l'utilisation des variables d'environnement en texte clair pour les informations sensibles, comme les données d'identification.

`name`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Nom de la variable d'environnement.

`value`

Type : String

Obligatoire : oui lorsque `environment` est utilisé

Valeur de la variable d'environnement.

```
"environment" : [
    { "name" : "string", "value" : "string" },
    { "name" : "string", "value" : "string" }
]
```

`secrets`

Type : tableau d'objets

Obligatoire : non

Objet représentant le secret à exposer à votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer au conteneur. Les valeurs prises en charge sont soit le nom de ressource Amazon complet (ARN), soit AWS Secrets Manager secret ou l'ARN complet du paramètre dans le AWS Systems Manager Magasin de paramètres.

> **Note**
>
> Si le paramètre Systems Manager Parameter Store ou le paramètre Secrets Manager existe dans le même Région AWS comme tâche que vous lancez, vous pouvez utiliser soit l'ARN complet, soit le nom du secret. Si le paramètre existe dans une autre région, l'ARN complet doit être spécifié.

```
"secrets": [
    {
        "name": "environment_variable_name",
        "valueFrom": "arn:aws:ssm:region:aws_account_id:parameter/parameter_name"
    }
]
```

Paramètres réseau

`disableNetworking`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, la mise en réseau est désactivée dans le conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

La valeur par défaut est `false`.

```
"disableNetworking": true|false
```

`links`

Type : tableau de chaînes

Obligatoire : non

Le paramètre `link` permet aux conteneurs de communiquer entre eux sans avoir besoin de définir des mappages de ports. Ce paramètre n'est pris en charge que si le mode réseau d'une définition de tâche est défini sur `bridge`. La construction `name:internalName` est analogue à `name:alias` dans les liaisons Docker. Il peut comporter jusqu'à 255 lettres (majuscules et minuscules), des chiffres, des tirets ou des traits de soulignement.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

> **Important**
>
> Des conteneurs situés sur la même instance de conteneur peuvent communiquer entre eux sans nécessiter de liens ou de mappages de ports hôtes. L'isolation réseau sur une instance de conteneur est contrôlée par les groupes de sécurité et les paramètres de VPC.

```
"links": ["name:internalName", ...]
```

`hostname`

Type : chaîne

Obligatoire : non

Nom d'hôte à utiliser pour votre conteneur. Ce paramètre correspond à `Hostname` dans docker create-container et à l'option `--hostname` de docker run.

> **Note**
>
> Si vous utilisez le mode réseau `awsvpc`, le paramètre `hostname` n'est pas pris en charge.

```
"hostname": "string"
```

`dnsServers`

Type : tableau de chaînes

Obligatoire : non

Liste de serveurs DNS qui sont présentés au conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"dnsServers": ["string", ...]
```

`dnsSearchDomains`

Type : tableau de chaînes

Obligatoire : non

Motif : ^ [a-zA-Z0-9-.] {0,253} [a-zA-Z0-9] $

Liste des domaines de recherche DNS qui sont présentés au conteneur. Ce paramètre correspond à `DnsSearch` dans la commande docker create-container et à l'option `--dns-search` dans la commande docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"dnsSearchDomains": ["string", ...]
```

`extraHosts`

Type : tableau d'objets

Obligatoire : non

Liste de noms d'hôte et de mappages d'adresses IP à ajouter au fichier `/etc/hosts` dans le conteneur.

Ce paramètre correspond à `ExtraHosts` dans la commande docker create-container et à l'option `--add-host` dans la commande docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le mode réseau `awsvpc`.

```
"extraHosts": [
      {
        "hostname": "string",
        "ipAddress": "string"
      }
      ...
```

```
    ]
```

`hostname`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Nom d'hôte à utiliser dans l'entrée `/etc/hosts`.

`ipAddress`

Type : String

Obligatoire : oui, lorsque des objets `extraHosts` sont utilisés

Adresse IP à utiliser dans l'entrée `/etc/hosts`.

Stockage et journalisation

`readonlyRootFilesystem`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur ne dispose que d'un accès en lecture seule au système de fichiers racine. Ce paramètre correspond à `ReadonlyRootfs` dans la commande docker create-container et à l'option `--read-only` dans la commande docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

La valeur par défaut est `false`.

```
"readonlyRootFilesystem": true|false
```

`mountPoints`

Type : tableau d'objets

Obligatoire : non

Les points de montage pour les volumes de données dans votre conteneur. Ce paramètre correspond à `Volumes` dans l'API Docker create-container et à l'option `--volume` de docker run.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`. Les conteneurs Windows ne peuvent pas monter de répertoires sur un autre lecteur, et les points de montage ne peuvent pas être utilisés sur plusieurs lecteurs. Vous devez spécifier des points de montage pour associer un volume Amazon EBS directement à une tâche Amazon ECS.

`sourceVolume`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Nom du volume à monter.

`containerPath`

Type : String

Obligatoire : oui, lorsque des objets `mountPoints` sont utilisés

Le chemin dans le conteneur où le volume sera monté.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

Pour les tâches qui exécutent le système d'exploitation Windows, laissez la valeur par défaut de `false`.

`volumesFrom`

Type : tableau d'objets

Obligatoire : non

Volumes de données à monter à partir d'un autre conteneur. Ce paramètre correspond à `VolumesFrom` dans la commande docker create-container et à l'option `--volumes-from` dans la commande docker run.

`sourceContainer`

Type : String

Obligatoire : oui lorsque `volumesFrom` est utilisé

Nom du conteneur à partir duquel monter les volumes.

`readOnly`

Type : booléen

Obligatoire : non

Si cette valeur est `true`, le conteneur ne peut accéder au volume qu'en lecture. Si cette valeur est `false`, le conteneur peut écrire sur le volume. La valeur par défaut est `false`.

```
"volumesFrom": [
                {
                  "sourceContainer": "string",
                  "readOnly": true|false
                }
              ]
```

`logConfiguration`

Tapez : LogConfiguration Objet

Obligatoire : non

Spécification de configuration du journal pour le conteneur.

Pour obtenir des exemples de définitions de tâche qui utilisent une configuration de journaux, consultez Exemple de définitions de tâches Amazon ECS.

Ce paramètre correspond à `LogConfig` dans la commande docker create-container et à l'option `--log-driver` dans la commande docker run. Par défaut, les conteneurs utilisent le même pilote de journalisation que celui utilisé par le démon Docker. Cependant, le conteneur peut utiliser un pilote de journalisation différent que celui du démon Docker en spécifiant un pilote de

journal avec ce paramètre dans la définition de conteneur. Pour utiliser un pilote de journalisation différent pour un conteneur, le système de journal doit être configuré correctement sur l'instance de conteneur (ou sur un serveur de journal différent pour les options de journalisation à distance).

Tenez compte des éléments suivants lorsque vous spécifiez une configuration de journal pour vos conteneurs :

- Amazon ECS prend en charge un sous-ensemble des pilotes de journalisation qui sont disponibles pour le démon Docker.
- Ce paramètre nécessite la version 1.18 ou ultérieure de l'API Docker à distance sur votre instance de conteneur.
- L'agent de conteneur Amazon ECS qui s'exécute sur une instance de conteneur doit enregistrer les pilotes de journalisation disponibles sur cette instance avec la variable d'environnement `ECS_AVAILABLE_LOGGING_DRIVERS` avant que les conteneurs placés sur cette instance puissent utiliser ces options de configuration de journalisation. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

```
"logConfiguration": {
      "logDriver": "awslogs","fluentd","gelf","json-
file","journald","splunk","syslog","awsfirelens",
      "options": {"string": "string"
        ...},
 "secretOptions": [{
  "name": "string",
  "valueFrom": "string"
 }]
}
```

`logDriver`

Type : String

Valeurs valides : `"awslogs","fluentd","gelf","json-file","journald","splunk","syslog","awsfirelens"`

Obligatoire : oui lorsque `logConfiguration` est utilisé

Pilote de journal à utiliser pour le conteneur. Par défaut, les valeurs valides mentionnées ci-dessus sont les pilotes de journal avec lesquels l'agent de conteneur Amazon ECS peut communiquer.

Les pilotes de journal pris en charge sont `awslogs`, `fluentd`, `gelf`, `json-file`, `journald`, `syslog`, `splunk` et `awsfirelens`.

Pour plus d'informations sur l'utilisation du `awslogs` enregistrez le pilote dans les définitions de tâches pour envoyer les journaux de vos conteneurs à CloudWatch Journaux, voir Envoyer les journaux Amazon ECS à CloudWatch.

Pour de plus amples informations sur l'utilisation du pilote de journal `awsfirelens`, consultez Routage personnalisé des journaux.

> **Note**
>
> Si vous avez un pilote personnalisé qui n'est pas répertorié, vous pouvez bifurquer le projet d'agent de conteneur Amazon ECS qui est disponible sur GitHub et personnalisez-le pour qu'il fonctionne avec ce pilote. Nous vous conseillons d'envoyer des demandes d'extraction pour les modifications que vous souhaitez inclure. Toutefois, nous ne prenons pas en charge l'exécution de copies modifiées de ce logiciel pour le moment.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options de configuration à envoyer au pilote de journal.

Les options que vous pouvez spécifier dépendent du pilote de journalisation. Certaines des options que vous pouvez spécifier lorsque vous utilisez le `awslogs` routeur pour acheminer les journaux vers Amazon CloudWatch inclure les éléments suivants :

`awslogs-create-group`

Obligatoire : non

Spécifiez si vous souhaitez que le groupe de journaux soit créé automatiquement. Si cette option n'est pas spécifiée, l'emplacement par défaut est `false`.

> **Note**
>
> Votre politique IAM doit inclure l'autorisation `logs:CreateLogGroup` afin que vous puissiez essayer d'utiliser `awslogs-create-group`.

`awslogs-region`

Obligatoire : oui

Spécifiez le Région AWS que le `awslogs` log driver est destiné à envoyer vos journaux Docker à. Vous pouvez choisir d'envoyer tous vos journaux provenant de clusters de différentes régions vers une seule région dans CloudWatch Journaux. Cela leur permet d'être tous visibles en un seul endroit. Sinon, vous pouvez les séparer par région pour plus de granularité. Assurez-vous que le groupe de journaux spécifié existe dans la région que vous spécifiez avec cette option.

`awslogs-group`

Obligatoire : oui

Assurez-vous de spécifier un groupe de journaux auquel le pilote de journaux `awslogs` envoie ses flux de journaux.

`awslogs-stream-prefix`

Obligatoire : Facultatif

Utilisez l'option `awslogs-stream-prefix` pour associer un flux de journaux au préfixe spécifié, au nom du conteneur et à l'ID de la tâche Amazon ECS à laquelle appartient le conteneur. Si vous spécifiez un préfixe avec cette option, le format du flux de journaux est le suivant.

```
prefix-name/container-name/ecs-task-id
```

Si vous ne spécifiez pas de préfixe avec cette option, le flux de journaux est nommé d'après l'ID de conteneur attribué par le démon Docker sur l'instance de conteneur. Etant donné qu'il est difficile de suivre les journaux jusqu'au conteneur qui les a envoyé en utilisant uniquement l'ID de conteneur Docker (qui est uniquement disponible sur l'instance de conteneur), nous vous recommandons de spécifier un préfixe avec cette option.

Pour les services Amazon ECS, vous pouvez utiliser le nom du service comme préfixe. Ce faisant, vous pourrez suivre les flux de journaux et déterminer le service auquel appartient le conteneur, trouver le nom du conteneur qui les a envoyés, ainsi que l'ID de la tâche à laquelle le conteneur appartient.

Vous devez spécifier un préfixe de flux pour vos journaux pour qu'ils apparaissent dans le panneau des journaux lors de l'utilisation de la console Amazon ECS.

`awslogs-datetime-format`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes au format `strftime` Python. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Ce format peut, par exemple, servir à analyser une sortie comme une pile de vidage, laquelle pourrait, dans le cas contraire, être consignée en plusieurs entrées. Le modèle adéquat permet de la capturer dans une seule entrée.

Pour plus d'informations, consultez awslogs-datetime-format.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

`awslogs-multiline-pattern`

Obligatoire : non

Cette option définit un modèle de démarrage à plusieurs lignes à l'aide d'une expression régulière. Un message de journal est composé de plusieurs lignes : une ligne correspondant au modèle et les lignes suivantes qui ne correspondent pas au modèle. La ligne mise en correspondance est le délimiteur entre les messages de journalisation.

Pour plus d'informations, consultez awslogs-multiline-pattern.

Cette option est ignorée si `awslogs-datetime-format` est également configuré.

Vous ne pouvez pas configurer à la fois les options `awslogs-datetime-format` et `awslogs-multiline-pattern`.

> **Note**
>
> La journalisation multiligne effectue l'analyse et la mise en correspondance des expressions régulières de tous les messages de journalisation. Cela peut avoir un impact négatif sur les performances de journalisation.

Les options suivantes s'appliquent à tous les pilotes de journalisation pris en charge.

`mode`

Obligatoire : non

Valeurs valides : `non-blocking` | `blocking`

Cette option définit le mode de transmission des messages de journalisation depuis le conteneur vers le pilote de journalisation spécifié à l'aide de `logDriver`. Le mode de livraison que vous choisissez affecte la disponibilité de l'application lorsque le flux des journaux provenant du conteneur est interrompu.

Si vous utilisez le mode `blocking` et que le flux de journaux est interrompu, les appels provenant du code du conteneur à écrire dans les flux `stdout` et `stderr` seront bloqués. Le fil de journalisation de l'application se bloquera en conséquence. Cela peut empêcher l'application de répondre et entraîner l'échec de vérification de l'état du conteneur.

Si vous utilisez le mode `non-blocking`, les journaux du conteneur sont plutôt stockés dans une mémoire tampon intermédiaire configurée avec l'option `max-buffer-size`. Cela empêche l'application de ne plus répondre lorsque les journaux ne peuvent pas être envoyés. Nous vous recommandons d'utiliser ce mode si vous souhaitez garantir la disponibilité du service et si vous acceptez une certaine perte de journaux. Pour plus d'informations, consultez la section Prévention de la perte de journaux grâce au mode non bloquant dans le pilote de journalisation conteneur `awslogs`.

Vous pouvez définir une valeur de `mode` par défaut pour tous les conteneurs d'un conteneur dans une Région AWS spécifique en utilisant le paramètre de

compte `defaultLogDriverMode`. Si vous ne spécifiez pas l'option `mode` dans `logConfiguration` ou ne configurez pas les paramètres du compte, Amazon ECS utilisera par défaut le mode `non-blocking`. Pour plus d'informations sur ce paramètre de compte, consultez la section le Mode de pilote du journal par défaut.

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. La quantité totale de mémoire allouée au niveau de la tâche doit être supérieure à la quantité de mémoire allouée à tous les conteneurs, en plus du tampon mémoire du pilote de journalisation.

> **Note**
>
> Le 25 juin 2025, Amazon ECS a modifié le mode par défaut du pilote de journalisation, passant de `blocking` à `non-blocking` afin de privilégier la disponibilité des tâches plutôt que la journalisation. Pour continuer à utiliser le mode `blocking` après cette modification, effectuez l'une des opérations suivantes :
>
> - Définissez l'option `mode` dans votre définition de conteneur `logConfiguration` sur `blocking`.
> - Réglez le paramètre du compte `defaultLogDriverMode` sur `blocking`.

`max-buffer-size`

Obligatoire : non

Valeur par défaut : `10 m`

Quand le mode `non-blocking` est utilisé, l'option de journalisation `max-buffer-size` contrôle la taille de la mémoire tampon utilisée pour le stockage des messages intermédiaires. Assurez-vous de spécifier une taille de mémoire tampon adéquate en fonction de votre application. Lorsque la mémoire tampon est pleine, il est impossible de stocker d'autres journaux. Les journaux qui ne peuvent pas être stockés sont perdus.

Pour acheminer les journaux à l'aide du routeur de journaux `splunk`, vous devez spécifier un `splunk-token` et une `splunk-url`.

Lorsque vous utilisez le `awsfirelens` routeur de journaux pour acheminer les journaux vers un Service AWS ou AWS Partner Network destination pour le stockage et l'analyse des journaux, vous pouvez définir `log-driver-buffer-limit` option permettant de limiter le nombre de lignes de journal qui sont mises en mémoire tampon avant d'être envoyées au conteneur Log Router. Cela peut aider à résoudre le problème potentiel de perte de journal, car un débit élevé peut entraîner un épuisement de la mémoire pour le tampon à l'intérieur de Docker. Pour plus d'informations, consultez the section called "Configuration des journaux pour un débit élevé".

Les autres options que vous pouvez spécifier lors de l'utilisation d'`awsfirelens` pour acheminer les journaux dépendent de la destination. Lorsque vous exportez des journaux vers Amazon Data Firehose, vous pouvez spécifier Région AWS avec `region` et un nom pour le flux de journaux avec `delivery_stream`.

Lorsque vous exportez des journaux vers Amazon Kinesis Data Streams, vous pouvez spécifier la Région AWS avec `region` et un nom pour le flux de données avec `stream`.

Lorsque vous exportez des journaux vers Amazon OpenSearch Service, vous pouvez spécifier des options telles que `Name`, `Host` (OpenSearch Point de terminaison de service (sans protocole), `Port`, `Index`, `Type`, `Aws_auth`, `Aws_region`, `Suppress_Type_Name`, et `tls`.

Lorsque vous exportez des journaux vers Amazon S3, vous pouvez spécifier le compartiment à l'aide de l'option `bucket`. Vous pouvez également spécifier `region`, `total_file_size`, `upload_timeout` et `use_put_object` en tant qu'options.

Ce paramètre nécessite la version 1.19 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`secretOptions`

Type : tableau d'objets

Obligatoire : non

Objet qui représente le secret à transmettre à la configuration de journal. Les secrets utilisés dans la configuration du journal peuvent inclure un jeton d'authentification, un certificat ou une clé de chiffrement. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Valeur à définir comme variable d'environnement sur le conteneur.

`valueFrom`

Type : chaîne

Obligatoire : oui

Secret à exposer à la configuration de journal du conteneur.

```
"logConfiguration": {
 "logDriver": "splunk",
 "options": {
  "splunk-url": "https://cloud.splunk.com:8080",
  "splunk-token": "...",
  "tag": "...",
  ...
 },
 "secretOptions": [{
  "name": "splunk-token",
  "valueFrom": "/ecs/logconfig/splunkcred"
 }]
}
```

`firelensConfiguration`

Tapez : FirelensConfiguration Objet

Obligatoire : non

Le FireLens configuration du conteneur. Cette option est utilisée pour spécifier et configurer un routeur de journal pour les journaux de conteneur. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

```
{
    "firelensConfiguration": {
        "type": "fluentd",
        "options": {
            "KeyName": ""
        }
    }
```

```
}
```

`options`

Type : mappage chaîne/chaîne

Obligatoire : non

Le key/value carte des options à utiliser lors de la configuration du log router. Ce champ est facultatif et peut être utilisé pour spécifier un fichier de configuration personnalisé ou ajouter des métadonnées supplémentaires, telles que les détails de tâche, de définition de tâche, de cluster et d'instance de conteneur à l'événement de journal. Si spécifié, la syntaxe à utiliser est `"options":{"enable-ecs-log-metadata":"true|false","config-file-type:"s3|file","config-file-value":"arn:aws:s3:::amzn-s3-demo-bucket/fluent.conf|filepath"}`. Pour plus d'informations, consultez Exemple de définition de tâche Amazon ECS : acheminer les journaux vers FireLens.

`type`

Type : chaîne

Obligatoire : oui

Routeur de journal à utiliser. Les valeurs valides sont `fluentd` ou `fluentbit`.

Sécurité

Pour plus d'informations sur la sécurité des conteneurs, consultez la section Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS.

`credentialSpecs`

Type : tableau de chaînes

Obligatoire : non

Une liste d'ARN dans SSM ou Amazon S3 associée à un fichier de spécifications d'informations d'identification (`CredSpec`) qui configure le conteneur pour l'authentification Active Directory. Nous vous recommandons d'utiliser ce paramètre plutôt que les `dockerSecurityOptions`. Le nombre maximal d'ARN est de 1.

Il existe deux formats pour chaque ARN.

credentialspecdomainless:MyARN

Vous utilisez `credentialspecdomainless:MyARN` pour fournir un `CredSpec` avec une section supplémentaire pour un secret dans Secrets Manager. Vous fournissez les informations d'identification de connexion au domaine dans le secret.

Chaque tâche exécutée sur une instance de conteneur peut rejoindre différents domaines.

Vous pouvez utiliser ce format sans associer l'instance de conteneur à un domaine.

credentialspec:MyARN

Vous utilisez `credentialspec:MyARN` pour fournir un `CredSpec` pour un domaine unique.

Vous devez joindre l'instance de conteneur au domaine avant de démarrer toute tâche utilisant cette définition de tâche.

Dans les deux formats, remplacez `MyARN` par l'ARN dans SSM ou Amazon S3.

Le `credspec` doit fournir un ARN dans Secrets Manager pour un secret contenant le nom d'utilisateur, le mot de passe et le domaine auquel se connecter. Pour une meilleure sécurité, l'instance n'est pas jointe au domaine pour l'authentification sans domaine. Les autres applications de l'instance ne peuvent pas utiliser les informations d'identification sans domaine. Vous pouvez utiliser ce paramètre pour exécuter des tâches sur la même instance, même si les tâches doivent être jointes à différents domaines. Pour plus d'informations, veuillez consulter Utilisation de gMSA pour les conteneurs Windows et Utilisation de gMSAs pour les conteneurs Linux.

`privileged`

Type : booléen

Obligatoire : non

Quand ce paramètre a la valeur true, le conteneur dispose de droits élevés sur l'instance de conteneur hôte (similaire à l'utilisateur `root`). Nous vous déconseillons d'exécuter des conteneurs avec `privileged`. Dans la plupart des cas, vous pouvez spécifier les privilèges exacts dont vous avez besoin en utilisant les paramètres spécifiques au lieu d'utiliser `privileged`.

Ce paramètre correspond à `Privileged` dans la commande docker create-container et à l'option `--privileged` dans la commande docker run.

> **Note**
>
> Ce paramètre n'est pas pris en charge pour les conteneurs ou tâches Windows qui utilisent le type de lancement Fargate.

La valeur par défaut est `false`.

```
"privileged": true|false
```

`user`

Type : chaîne

Obligatoire : non

Utilisateur à utiliser à l'intérieur du conteneur. Ce paramètre correspond à `User` dans la commande docker create-container et à l'option `--user` dans la commande docker run.

> **Important**
>
> Lorsque vous exécutez des tâches en mode réseau `host`, n'exécutez pas de conteneurs à l'aide de l'utilisateur root (UID 0). Comme bonne pratique de sécurité, utilisez toujours un utilisateur non root.

Vous pouvez spécifier l'`user` à l'aide des formats suivants. Si vous spécifiez un UID ou GID, vous devez le spécifier en tant que nombre entier positif.

- `user`
- `user:group`
- `uid`
- `uid:gid`
- `user:gid`
- `uid:group`

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"user": "string"
```

`dockerSecurityOptions`

Type : tableau de chaînes

Valeurs valides : « no-new-privileges » | « appArmor:profile » | « label :*value*Spécification des informations d'identification « | » :*CredentialSpecFilePath*«

Obligatoire : non

Liste de chaînes pour fournir une configuration personnalisée pour plusieurs systèmes de sécurité.

Pour les tâches Linux, ce paramètre peut être utilisé pour référencer des étiquettes personnalisées pour SELinux et des systèmes de sécurité à plusieurs niveaux AppArmor .

Ce paramètre peut être utilisé pour référencer un fichier de spécification d'informations d'identification qui configure un conteneur pour l'authentification Active Directory. Pour plus d'informations, consultez Utilisation des gMSA pour les conteneurs Windows EC2 pour Amazon ECS et Utilisation gMSA pour EC2 Linux conteneurs sur Amazon ECS.

Ce paramètre correspond à `SecurityOpt` dans la commande docker create-container et à l'option `--security-opt` dans la commande docker run.

```
"dockerSecurityOptions": ["string", ...]
```

> **Note**
>
> L'agent de conteneur Amazon ECS en cours d'exécution sur une instance de conteneur doit s'enregistrer avec les variables d'environnement `ECS_SELINUX_CAPABLE=true` ou `ECS_APPARMOR_CAPABLE=true` pour que les conteneurs placés sur cette instance puissent utiliser ces options de sécurité. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Limites des ressources

`ulimits`

Type : tableau d'objets

Obligatoire : non

Une liste de valeurs `ulimit` à définir pour un conteneur. Cette valeur remplace le paramètre de quota de ressources par défaut pour le système d'exploitation. Ce paramètre correspond à `Ulimits` dans la commande docker create-container et à l'option `--ulimit` dans la commande docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"ulimits": [
      {
        "name":
 "core"|"cpu"|"data"|"fsize"|"locks"|"memlock"|"msgqueue"|"nice"|"nofile"|"nproc"|"rss"|"rtp
        "softLimit": integer,
        "hardLimit": integer
      }
      ...
    ]
```

`name`

Type : String

Valeurs valides : `"core" | "cpu" | "data" | "fsize" | "locks" | "memlock" | "msgqueue" | "nice" | "nofile" | "nproc" | "rss" | "rtprio" | "rttime" | "sigpending" | "stack"`

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

Le `type` d'`ulimit`.

`hardLimit`

Type : Integer

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite stricte du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en nombre, selon le `type` d'`ulimit`.

`softLimit`

Type : Integer

Obligatoire : oui, lorsque des objets `ulimits` sont utilisés

La limite flexible du type `ulimit`. La valeur peut être spécifiée en octets, en secondes ou en nombre, selon le `type` d'`ulimit`.

Étiquettes Docker

`dockerLabels`

Type : mappage chaîne/chaîne

Obligatoire : non

UNE key/value carte des étiquettes à ajouter au contenant. Ce paramètre correspond à `Labels` dans la commande docker create-container et à l'option `--label` dans la commande docker run.

Ce paramètre nécessite la version 1.18 de l'API Docker Remote ou une version ultérieure sur votre instance de conteneur.

```
"dockerLabels": {"string": "string"
      ...}
```

Autres paramètres de définition de conteneur

Les paramètres de définition de conteneur suivants peuvent être utilisés lors de l'enregistrement des définitions de tâche dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Paramètres Linux
- Dépendances du conteneur
- Temporisations de conteneurs
- Contrôles système
- Interactive
- Pseudo Terminal

Paramètres Linux

`linuxParameters`

Type : objet LinuxParameters

Obligatoire : non

Linux-les options spécifiques appliquées au conteneur, telles que KernelCapabilities.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"linuxParameters": {
      "capabilities": {
        "add": ["string", ...],
        "drop": ["string", ...]
        }
      }
```

`capabilities`

Type : objet KernelCapabilities

Obligatoire : non

Caractéristiques Linux du conteneur ajoutées ou supprimées de la configuration par défaut fournie par Docker. Pour plus d'informations sur ces caractéristiques Linux, veuillez consulter

la page consacrée aux caractéristiques(7) dans le manuel Linux (langue française non garantie).

`add`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_READ" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" | "NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW" | "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" | "SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT" | "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" | "SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond à `CapAdd` dans la commande docker create-container et à l'option `--cap-add` dans la commande docker run.

`add`

Type : tableau de chaînes

Valeurs valides : `"SYS_PTRACE"`

Obligatoire : non

Caractéristiques Linux du conteneur à ajouter à la configuration par défaut fournie par Docker. Ce paramètre correspond à `CapAdd` dans la commande docker create-container et à l'option `--cap-add` dans la commande docker run.

`drop`

Type : tableau de chaînes

Valeurs valides : `"ALL" | "AUDIT_CONTROL" | "AUDIT_WRITE" | "BLOCK_SUSPEND" | "CHOWN" | "DAC_OVERRIDE" | "DAC_READ_SEARCH" | "FOWNER" | "FSETID" | "IPC_LOCK" | "IPC_OWNER" | "KILL" | "LEASE" | "LINUX_IMMUTABLE" | "MAC_ADMIN" | "MAC_OVERRIDE" | "MKNOD" |`

```
"NET_ADMIN" | "NET_BIND_SERVICE" | "NET_BROADCAST" | "NET_RAW"
| "SETFCAP" | "SETGID" | "SETPCAP" | "SETUID" | "SYS_ADMIN" |
"SYS_BOOT" | "SYS_CHROOT" | "SYS_MODULE" | "SYS_NICE" | "SYS_PACCT"
| "SYS_PTRACE" | "SYS_RAWIO" | "SYS_RESOURCE" | "SYS_TIME" |
"SYS_TTY_CONFIG" | "SYSLOG" | "WAKE_ALARM"
```

Obligatoire : non

Caractéristiques Linux du conteneur à supprimer de la configuration par défaut fournie par Docker. Ce paramètre correspond à `CapDrop` dans la commande docker create-container et à l'option `--cap-drop` dans la commande docker run.

`devices`

Tous les périphériques hôtes à exposer au conteneur. Ce paramètre correspond à `Devices` dans la commande docker create-container et `--device` option pour exécuter Docker.

Type : tableau d'objets Périphérique

Obligatoire : non

`hostPath`

Chemin du périphérique dans l'instance de conteneur hôte.

Type : chaîne

Obligatoire : oui

`containerPath`

Chemin auquel exposer l'appareil hôte à l'intérieur du conteneur.

Type : chaîne

Obligatoire : non

`permissions`

Autorisations explicites à fournir au conteneur pour le périphérique. Par défaut, le conteneur dispose des autorisations `read`, `write` et `mknod` sur l'appareil.

Type : tableau de chaînes

Valeurs valides : `read` | `write` | `mknod`

`initProcessEnabled`

Exécutez un processus `init` dans le conteneur afin de transmettre des signaux et de récolter les processus. Ce paramètre correspond à l'option `--init` de docker run.

Ce paramètre nécessite la version 1.25 de l'API Docker à distance ou une version supérieure sur votre instance de conteneur.

`maxSwap`

Quantité totale de mémoire d'échange (en Mio) qu'un conteneur peut utiliser. Ce paramètre est converti en l'option `--memory-swap` de la commande docker run, où la valeur est la somme de la mémoire du conteneur et de la valeur `maxSwap`.

Si la valeur `0` est spécifiée pour `maxSwap`, le conteneur n'utilise pas l'échange. Les valeurs acceptées sont `0` ou n'importe quel nombre entier positif. Si le paramètre `maxSwap` n'est pas spécifié, le conteneur utilise la configuration d'échange pour l'instance de conteneur sur laquelle il s'exécute. Une valeur `maxSwap` doit être définie pour que le paramètre `swappiness` soit utilisé.

`sharedMemorySize`

Valeur correspondant à la taille (en Mio) du volume `/dev/shm`. Ce paramètre correspond à l'option `--shm-size` de docker run.

Type : Integer

`swappiness`

Vous pouvez utiliser ce paramètre pour régler le comportement d'échange de mémoire d'un conteneur. Une valeur `swappiness` de `0` empêche l'échange de se produire à moins que cela ne soit nécessaire. Une valeur `swappiness` de `100` entraîne fréquemment l'échange de pages. Les valeurs acceptées sont les nombres entiers compris entre `0` et `100`. Si vous ne spécifiez aucune valeur, la valeur par défaut `60` est utilisée. De plus, si vous ne spécifiez pas de valeur pour `maxSwap`, ce paramètre est ignoré. Ce paramètre correspond à l'option `--memory-swappiness` de docker run.

> **Note**
>
> Si vous utilisez des tâches sur Amazon Linux 2023, le paramètre `swappiness` n'est pas pris en charge.

`tmpfs`

Chemin du conteneur, options de montage et taille maximale (en Mio) du montage tmpfs. Ce paramètre correspond à l'option `--tmpfs` de docker run.

Type : tableau d'objets Tmpfs

Obligatoire : non

`containerPath`

Chemin de fichier absolu où sera monté le volume tmpfs.

Type : chaîne

Obligatoire : oui

`mountOptions`

Liste des options de montage du volume tmpfs.

Type : tableau de chaînes

Obligatoire : non

Valeurs valides : `"defaults" | "ro" | "rw" | "suid" | "nosuid" | "dev" | "nodev" | "exec" | "noexec" | "sync" | "async" | "dirsync" | "remount" | "mand" | "nomand" | "atime" | "noatime" | "diratime" | "nodiratime" | "bind" | "rbind" | "unbindable" | "runbindable" | "private" | "rprivate" | "shared" | "rshared" | "slave" | "rslave" | "relatime" | "norelatime" | "strictatime" | "nostrictatime" | "mode" | "uid" | "gid" | "nr_inodes" | "nr_blocks" | "mpol"`

`size`

Taille maximale (en Mio) du volume tmpfs.

Type : Integer

Obligatoire : oui

Dépendances du conteneur

`dependsOn`

Type : tableau d'objets ContainerDependency

Obligatoire : non

Dépendances définies pour le démarrage et l'arrêt de conteneurs. Un conteneur peut contenir plusieurs dépendances. Lorsqu'une dépendance est définie pour le démarrage des conteneurs, elle est inversée pour leur arrêt. Pour obtenir un exemple, consultez Dépendances du conteneur.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

Les instances nécessitent au moins la version `1.26.0` de l'agent de conteneur pour activer les dépendances de conteneur. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez un Amazon ECS-optimized Amazon Linux AMI, votre instance a besoin d'au moins une version `1.26.0-1` de la `ecs-init` colis. Si vos instances de conteneur sont lancées à partir de la version `20190301` ou ultérieure, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

```
"dependsOn": [
    {
        "containerName": "string",
        "condition": "string"
    }
]
```

`containerName`

Type : chaîne

Obligatoire : oui

Nom de conteneur qui doit répondre à la condition spécifiée.

`condition`

Type : chaîne

Obligatoire : oui

Condition de dépendance du conteneur. Voici les conditions disponibles et leur comportement :

- `START` - Cette condition émule le comportement de liens et de volumes aujourd'hui. La condition vérifie qu'un conteneur dépendant est démarré avant d'autoriser d'autres conteneurs à démarrer.
- `COMPLETE` - Cette condition vérifie qu'un conteneur dépendant exécute un cycle complet (sorties) avant d'autoriser d'autres conteneurs à démarrer. Cela peut être utile pour les conteneurs non essentiels qui exécutent un script, puis se ferment. Cette condition ne peut pas être définie sur un conteneur essentiel.
- `SUCCESS` - Cette condition est identique à `COMPLETE`, mais elle nécessite également que le conteneur se termine par un état `zero`. Cette condition ne peut pas être définie sur un contenant essentiel.
- `HEALTHY` - Cette condition vérifie que la surveillance de l'état du conteneur dépendant réussit avant d'autoriser d'autres conteneurs à démarrer. Cela nécessite que le conteneur dépendant dispose de surveillances de l'état configurées dans la définition de la tâche. Cette condition n'est confirmée qu'au démarrage de la tâche.

Temporisations de conteneurs

`startTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée d'attente (en secondes) avant l'abandon de la résolution de dépendances pour un conteneur.

Par exemple, vous spécifiez deux conteneurs dans une définition de tâche, où le `containerA` dispose d'une dépendance au `containerB` avec un état `COMPLETE`, `SUCCESS` ou `HEALTHY`. Si

une valeur `startTimeout` est spécifiée pour `containerB` et qu'il n'a pas atteint l'état souhaité dans ce délai, alors le `containerA` ne démarre pas.

> **Note**
>
> Si un conteneur ne respecte pas une contrainte de dépendance ou s'écoule avant de respecter la contrainte, Amazon ECS ne fait pas progresser pas les conteneurs dépendants vers leur état suivant.

La valeur maximale est de 120 secondes.

`stopTimeout`

Type : Integer

Obligatoire : non

Exemples de valeur : `120`

Durée (en secondes) à attendre avant que le conteneur soit expressément tué s'il ne s'achève pas normalement tout seul.

Si le paramètre `stopTimeout` n'est pas spécifié, la valeur définie pour la variable de configuration `ECS_CONTAINER_STOP_TIMEOUT` de l'agent de conteneur Amazon ECS est utilisée. Si ni le paramètre `stopTimeout` ni la variable de configuration de l'agent `ECS_CONTAINER_STOP_TIMEOUT` ne sont définis, les valeurs par défaut de 30 secondes pour les conteneurs Linux et de 30 secondes pour les conteneurs Windows sont utilisées. Les instances de conteneur doivent au moins disposer de la version 1.26.0 de l'agent de conteneur pour permettre une valeur de temporisation d'arrêt d'un conteneur. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez un Amazon ECS-optimized Amazon Linux AMI, votre instance nécessite au moins la version 1.26.0-1 du `ecs-init` colis. Si vos instances de conteneur sont lancées à partir de la version `20190301` ou ultérieure, elles contiennent les versions requises de l'agent de conteneur et `ecs-init`. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

Contrôles système

`systemControls`

Type : objet SystemControl

Obligatoire : non

Liste des paramètres du noyau de l'espace de noms à définir dans le conteneur. Ce paramètre correspond à `Sysctls` dans la commande docker create-container et à l'option `--sysctl` dans la commande docker run. Par exemple, vous pouvez configurer le paramètre `net.ipv4.tcp_keepalive_time` pour maintenir des connexions de plus longue durée.

Il n'est pas recommandé de spécifier des paramètres `systemControls` liés au réseau pour plusieurs conteneurs dans une seule tâche qui utilise également le mode réseau `awsvpc` ou `host`. En procédant ainsi, vous vous exposez aux inconvénients suivants :

- Pour les tâches qui utilisent le mode réseau `awsvpc`, si vous définissez le paramètre `systemControls` pour un conteneur, il s'applique à tous les conteneurs de la tâche. Si vous définissez différents paramètres `systemControls` pour plusieurs conteneurs dans une seule tâche, c'est le conteneur qui est démarré en dernier qui détermine le paramètre `systemControls` qui prend effet.
- Les définitions de tâche qui utilisent le mode réseau `host`, l'espace de noms `systemControls` ne sont pas prises en charge.

Si vous définissez un espace de noms des ressources IPC à utiliser pour les conteneurs de la tâche, les conditions suivantes s'appliquent aux contrôles de système. Pour plus d'informations, consultez Mode IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les valeurs des `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"systemControls": [
```

```
    {
         "namespace":"string",
         "value":"string"
    }
]
```

`namespace`

Type : chaîne

Obligatoire : non

Paramètre du noyau de l'espace de noms pour lequel définir une `value`.

Valeurs d'espace de noms IPC valides : `"kernel.msgmax" | "kernel.msgmnb" | "kernel.msgmni" | "kernel.sem" | "kernel.shmall" | "kernel.shmmax" | "kernel.shmmni" | "kernel.shm_rmid_forced"`, et `Sysctls` qui commencent par `"fs.mqueue.*"`

Valeurs d'espace de noms de réseau valides : `Sysctls` qui commencent par `"net.*"`

`value`

Type : chaîne

Obligatoire : non

Valeur du paramètre du noyau de l'espace de noms spécifié dans `namespace`.

Interactive

`interactive`

Type : booléen

Obligatoire : non

Lorsque ce paramètre est `true`, vous pouvez déployer des applications conteneurisées qui nécessitent l'allocation d'une `stdin` ou d'un `tty`. Ce paramètre correspond à `OpenStdin` dans la commande docker create-container et à l'option `--interactive` dans la commande docker run.

La valeur par défaut est `false`.

Pseudo Terminal

`pseudoTerminal`

Type : booléen

Obligatoire : non

Lorsque ce paramètre a la valeur `true`, un TTY est alloué. Ce paramètre correspond à `Tty` dans la commande docker create-container et à l'option `--tty` dans la commande docker run.

La valeur par défaut est `false`.

Nom de l'accélérateur Elastic Inference (obsolète)

L'exigence de la ressource d'accélérateur Elastic Inference pour votre définition de tâche.

> **Note**
>
> Amazon Elastic Inference (EI) n'est plus disponible pour les clients.

Les paramètres suivants sont autorisés dans une définition de tâche :

`deviceName`

Type : chaîne

Obligatoire : oui

Nom de l'appareil accélérateur d'inférence élastique pour l'instance. `deviceName` doit également être référencé dans une définition du conteneur. Consultez `Elastic Inference accelerator`.

`deviceType`

Type : chaîne

Obligatoire : oui

L'accélérateur Elastic Inference à utiliser.

Contraintes de placement des tâches

Lorsque vous enregistrez une définition de tâche, vous pouvez fournir des contraintes de placement des tâches qui personnalisent la façon dont Amazon ECS place les tâches.

Vous pouvez utiliser des contraintes pour placer des tâches en fonction de la zone de disponibilité, du type d'instance ou d'attributs personnalisés. Pour plus d'informations, consultez Définition des instances de conteneur utilisées par Amazon ECS pour les tâches.

Les paramètres suivants sont autorisés dans une définition de conteneur:

`expression`

Type : chaîne

Obligatoire : non

Expression de langage de requête de cluster à appliquer à la contrainte. Pour plus d'informations, consultez Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

`type`

Type : chaîne

Obligatoire : oui

Type de contrainte. Utilisez `memberOf` pour limiter la sélection à un groupe de candidats valides.

Configuration du proxy

`proxyConfiguration`

Type : objet ProxyConfiguration

Obligatoire : non

Détails de configuration pour le proxy App Mesh.

Pour les tâches qui utilisent EC2, les instances de conteneur nécessitent au moins la version 1.26.0 de l'agent de conteneur et au moins la version 1.26.0-1 du package `ecs-init` pour activer une configuration proxy. Si vos instances de conteneurs sont lancées depuis Amazon ECS-optimized Version AMI `20190301` ou une version ultérieure, ils contiennent les versions

requises de l'agent de conteneur et `ecs-init`. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

```
"proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "string",
    "properties": [
        {
           "name": "string",
           "value": "string"
        }
    ]
}
```

`type`

Type : String

Valeur valeurs : `APPMESH`

Obligatoire : non

Type de proxy. La seule valeur prise en charge est `APPMESH`.

`containerName`

Type : chaîne

Obligatoire : oui

Nom du conteneur qui fait office de proxy App Mesh.

`properties`

Type : tableau d'objets KeyValuePair

Obligatoire : non

L'ensemble de paramètres de configuration réseau pour fournir le plug-in Container Network Interface (CNI), spécifié sous la forme de paires clé-valeur.

- `IgnoredUID` – (Obligatoire) ID d'utilisateur (UID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredGID`, ce champ doit être vide.
- `IgnoredGID` – (Obligatoire) ID de groupe (GID) du conteneur de proxy, tel que défini par le paramètre `user` dans une définition de conteneur. Ce port est utilisé pour garantir que le proxy ignore son propre trafic. Si vous avez spécifié `IgnoredUID`, ce champ doit être vide.
- `AppPorts` - (Obligatoire) Liste des ports utilisés par l'application. Le trafic réseau vers ces ports est transmis à `ProxyIngressPort` et `ProxyEgressPort`.
- `ProxyIngressPort` – (Obligatoire) Spécifie le port auquel le trafic entrant vers `AppPorts` est dirigé.
- `ProxyEgressPort` – (Obligatoire) Spécifie le port auquel le trafic sortant de `AppPorts` est dirigé.
- `EgressIgnoredPorts` – (Obligatoire) Le trafic sortant accédant à ces ports spécifiés est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.
- `EgressIgnoredIPs` – (Obligatoire) Le trafic sortant accédant à ces adresses IP spécifiées est ignoré et n'est pas redirigé vers le `ProxyEgressPort`. Cela peut être une liste vide.

`name`

Type : chaîne

Obligatoire : non

Nom de la paire clé-valeur.

`value`

Type : chaîne

Obligatoire : non

Valeur de la paire clé-valeur.

Volumes

Lorsque vous enregistrez une définition de tâche, vous pouvez spécifier une liste de volumes à transmettre au démon Docker sur une instance de conteneur, à laquelle d’autres conteneurs de la même instance de conteneur pourront accéder.

Les types de volumes de données qui peuvent être utilisés sont les suivants :

- Volumes Amazon EBS : fournit un stockage par blocs rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données. Vous pouvez attacher un volume Amazon EBS par tâche Amazon ECS lorsque vous exécutez une tâche autonome ou lorsque vous créez ou mettez à jour un service. Les volumes Amazon EBS sont pris en charge pour les tâches Linux. Pour plus d'informations, consultez Utilisation des volumes Amazon EBS avec Amazon ECS.
- Volumes Amazon EFS : fournit un stockage de fichiers simple, évolutif et permanent à utiliser avec vos tâches Amazon ECS. Avec Amazon EFS, la capacité de stockage est élastique. Elle augmente et diminue automatiquement au fil de vos ajouts et suppressions de fichiers. Vos applications peuvent disposer du stockage dont elles ont besoin, au moment où elles en ont besoin. Les volumes Amazon EFS sont pris en charge. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.
- Volumes FSx for Windows File Server : fournit des serveurs de fichiers Microsoft Windows entièrement gérés. Ces serveurs de fichiers sont basés sur un système de fichiers Windows. Lorsque vous utilisez FSx for Windows File Server avec Amazon ECS, vous pouvez allouer vos tâches Windows avec un stockage de fichiers permanent, distribué, partagé et statique. Pour plus d'informations, consultez Utilisation de FSx for Windows File Server avec Amazon ECS.

  Les conteneurs Windows sur Fargate ne prennent pas en charge cette option.
- Volumes Docker — A Docker-managed volume créé sous `/var/lib/docker/volumes` sur l'instance Amazon EC2 hôte. Les pilotes de volume Docker (également appelés plugins) sont utilisés pour intégrer les volumes à des systèmes de stockage externes, tels qu'Amazon EBS. Le pilote de volume `local` intégré ou un pilote de volume tiers peut être utilisé. Les volumes Docker sont pris en charge uniquement lors de l'exécution de tâches sur des instances Amazon EC2. Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`. Pour utiliser des volumes Docker, spécifiez un `dockerVolumeConfiguration` dans votre définition de tâche.
- Montages liés : un fichier ou un dossier sur la machine hôte est monté dans un conteneur. Les volumes hôtes à montage lié sont pris en charge. Pour utiliser des volumes hôte de montage lié, spécifiez un `host` et éventuellement une valeur `sourcePath` dans votre définition de tâche.

Pour plus d'informations, consultez Options de stockage pour les tâches Amazon ECS.

Les paramètres suivants sont autorisés dans une définition de conteneur.

`name`

Type : chaîne

Obligatoire : non

Nom du volume. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union (`-`) et traits de soulignement (`_`) sont autorisés. Ce nom est référencé dans le paramètre `sourceVolume` de l'objet `mountPoints` de définition du conteneur.

`host`

Obligatoire : non

Le paramètre `host` est utilisé pour lier le cycle de vie du montage lié à l'instance Amazon EC2 hôte, plutôt qu'à la tâche et à l'endroit où elle est stockée. Si le paramètre `host` est vide, le démon Docker attribue un chemin hôte au volume de données, mais la persistance des données après l'arrêt des conteneurs qui lui sont associés n'est pas garantie.

Les conteneurs Windows peuvent monter des répertoires entiers sur le même lecteur que `$env:ProgramData`.

> **Note**
>
> Le `sourcePath` Le paramètre est pris en charge uniquement lors de l'utilisation de tâches hébergées sur des instances Amazon EC2 ou des instances gérées Amazon ECS.

`sourcePath`

Type : chaîne

Obligatoire : non

Lorsque le paramètre `host` est utilisé, spécifiez un paramètre `sourcePath` pour déclarer le chemin d'accès sur l'instance Amazon EC2 hôte qui est présentée au conteneur. Si ce paramètre est vide, le démon Docker attribue un chemin hôte pour vous. Si le paramètre `host` contient un emplacement de fichier `sourcePath`, le volume de données persiste à l'emplacement spécifié sur l'instance Amazon EC2 hôte jusqu'à ce que vous le supprimiez manuellement. Si la valeur `sourcePath` n'existe pas sur l'instance Amazon EC2 hôte, le démon Docker la crée. Si l'emplacement n'existe pas, le contenu du chemin source est exporté.

`configuredAtLaunch`

Type : booléen

Obligatoire : non

Spécifie si un volume est configurable au lancement. Lorsqu'il est défini sur `true`, vous pouvez configurer le volume lors de l'exécution d'une tâche autonome ou lors de la création ou de la mise à jour d'un service. Lorsque cette option est définie sur `true`, vous ne pourrez pas fournir d'autre configuration de volume dans la définition de tâche. Ce paramètre doit être défini sur `true` pour configurer un volume Amazon EBS à rattacher à une tâche. En définissant `configuredAtLaunch` sur `true` et en reportant la configuration du volume à la phase de lancement, vous pouvez créer des définitions de tâches qui ne sont pas limitées à un type de volume ou à des paramètres de volume spécifiques. Cela rend votre définition de tâche réutilisable dans différents environnements d'exécution. Pour plus d'informations, consultez Volumes Amazon EBS.

`dockerVolumeConfiguration`

Tapez : DockerVolumeConfiguration Objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Docker. Les volumes Docker sont pris en charge uniquement lors de l'exécution de tâches sur des instances EC2. Les conteneurs Windows prennent uniquement en charge l'utilisation du pilote `local`. Pour utiliser des montages liés, spécifiez plutôt un paramètre `host`.

`scope`

Type : String

Valeurs valides : `task` | `shared`

Obligatoire : non

Portée du volume Docker, qui détermine son cycle de vie. Les volumes Docker destinés à un élément `task` sont automatiquement mis en service lorsque la tâche commence, et détruits lorsque la tâche s'arrête. Les volumes Docker définis comme `shared` ne sont pas supprimés lorsque la tâche s'arrête.

`autoprovision`

Type : Boolean

Valeur par défaut : `false`

Obligatoire : non

Si cette valeur est `true`, le volume Docker est créé s'il n'existe pas déjà. Ce champ n’est utilisé que si `scope` a la valeur `shared`. Si le champ `scope` a la valeur `task`, ce paramètre doit être omis.

`driver`

Type : chaîne

Obligatoire : non

Pilote de volume Docker à utiliser. La valeur du pilote doit correspondre au nom du pilote fourni par Docker, car ce nom est utilisé pour le placement des tâches. Si le pilote a été installé à l’aide de l’interface CLI du plug-in Docker, utilisez `docker plugin ls` pour récupérer le nom du pilote à partir de votre instance de conteneur. Si le pilote a été installé à l’aide d’une autre méthode, utilisez la découverte du plug-in Docker pour récupérer le nom du pilote.

`driverOpts`

Type : chaîne

Obligatoire : non

Mappage des options spécifiques au pilote Docker à transmettre. Ce paramètre correspond à `DriverOpts` dans la section « Créer un volume » de Docker.

`labels`

Type : chaîne

Obligatoire : non

Métadonnées personnalisées à ajouter à votre volume Docker.

`efsVolumeConfiguration`

Tapez : EFSVolumeConfiguration Objet

Obligatoire : non

Ce paramètre est spécifié lorsque vous utilisez des volumes Amazon EFS.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers Amazon EFS à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : non

Répertoire du système de fichiers Amazon EFS à monter en tant que répertoire racine à l'intérieur de l'hôte. Si ce paramètre est omis, la racine du volume Amazon EFS est utilisée. La spécification de `/` a le même effet que l'omission de ce paramètre.

> **Important**
>
> Si un point d'accès EFS est spécifié dans `authorizationConfig`, le paramètre de répertoire racine doit être omis ou défini sur `/`, ce qui imposera le chemin défini sur le point d'accès EFS.

`transitEncryption`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique si vous souhaitez activer ou non le chiffrement des données Amazon EFS en transit entre l'hôte Amazon ECS et le serveur Amazon EFS. Si l'autorisation Amazon EFS IAM est utilisée, le chiffrement de transit doit être activé. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Chiffrement des données en transit dans le Guide de l'utilisateur Amazon Elastic File System.

`transitEncryptionPort`

Type : Integer

Obligatoire : non

Port à utiliser lors de l'envoi de données chiffrées entre l'hôte Amazon ECS et le serveur Amazon EFS. Si vous ne spécifiez pas de port de chiffrement en transit, la tâche utilisera la stratégie de sélection de port adoptée par l'assistant de montage Amazon EFS. Pour plus

d'informations, consultez Assistant de montage EFS dans le Guide de l'utilisateur Amazon Elastic File System User.

`authorizationConfig`

Tapez : EFSAuthorizationConfig Objet

Obligatoire : non

Détails de configuration des autorisations pour le système de fichiers Amazon EFS.

`accessPointId`

Type : chaîne

Obligatoire : non

ID du point d'accès à utiliser. Si un point d’accès est spécifié, la valeur du répertoire racine dans `efsVolumeConfiguration` doit être omise ou définie sur `/`, ce qui imposera le chemin défini sur le point d’accès EFS. Si un point d'accès est utilisé, le chiffrement de transit doit être activé dans `EFSVolumeConfiguration`. Pour plus d'informations, consultez Utilisation des points d'accès Amazon EFS dans le Guide de l'utilisateur Amazon Elastic File System.

`iam`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Indique s’il faut ou non utiliser le rôle IAM de tâche Amazon ECS spécifié dans une définition de tâche lors du montage du système de fichiers Amazon EFS. Si cette option est activée, le chiffrement en transit doit être activé dans la configuration `EFSVolumeConfiguration`. Si ce paramètre est omis, la valeur par défaut `DISABLED` est utilisée. Pour plus d'informations, consultez Rôles IAM pour les tâches.

`FSxWindowsFileServerVolumeConfiguration`

Tapez : FSxWindowsFileServerVolumeConfiguration Objet

Obligatoire : oui

Ce paramètre est spécifié lorsque vous utilisez un système de fichiers Amazon FSx for Windows File Server pour le stockage des tâches.

`fileSystemId`

Type : chaîne

Obligatoire : oui

ID du système de fichiers FSx for Windows File Server à utiliser.

`rootDirectory`

Type : chaîne

Obligatoire : oui

Répertoire du système de fichiers FSx for Windows File Server à monter en tant que répertoire racine à l'intérieur de l'hôte.

`authorizationConfig`

`credentialsParameter`

Type : chaîne

Obligatoire : oui

Options d'informations d'identification d'autorisation.

Options :

- Amazon Resource Name (ARN) d’un secret AWS Secrets Manager.
- ARN d’un paramètre AWS Systems Manager.

`domain`

Type : chaîne

Obligatoire : oui

Nom de domaine complet hébergé par un répertoire AWS Directory Service for Microsoft Active Directory (AWS Managed Microsoft AD) ou un Active Directory EC2 autohébergé.

Étiquettes

Lorsque vous enregistrez une définition de tâche, vous pouvez éventuellement spécifier des étiquettes de métadonnées qui sont appliquées à la définition de tâche. Les balises vous aident à classer et à organiser votre définition de tâche. Chaque balise est constituée d’une clé et d’une valeur

facultative. Vous définissez ces deux éléments. Pour plus d'informations, consultez Balisage des ressources Amazon ECS.

> **Important**
>
> N'ajoutez pas de données d'identification personnelle ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreuses personnes AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Les paramètres suivants sont autorisés dans un objet balise.

`key`

Type : chaîne

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : chaîne

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

Autres paramètres de définition de tâche

Les paramètres de définition de tâche suivants peuvent être utilisés lors de l'enregistrement des définitions de tâches dans la console Amazon ECS à l'aide de l'option Configure via JSON (Configurer via JSON). Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Rubriques

- Mode IPC
- Mode PID

- Injection de pannes

Mode IPC

`ipcMode`

Type : chaîne

Obligatoire : non

Espace de noms de ressource IPC à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host`, `task` ou `none`. Si `host` est spécifié, tous les conteneurs des tâches ayant spécifié le mode IPC `host` sur la même instance de conteneur partagent les mêmes ressources IPC avec l'instance Amazon EC2 hôte. Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent les mêmes ressources IPC. Si `none` est spécifié, les ressources IPC des conteneurs d'une tâche sont privés et ne partagent rien avec les autres conteneurs d'une tâche ou d'une instance de conteneur. Si aucune valeur n'est spécifiée, le partage de l'espace de noms de ressource IPC dépend du paramètre du démon Docker sur l'instance de conteneur.

Si le mode IPC `host` est utilisé, il existe un risque accru d'exposition à un espace de noms IPC indésirable.

Si vous définissez les paramètres du noyau placés dans un espace de noms à l'aide de `systemControls` pour les conteneurs de la tâche, les éléments suivants s'appliquent à votre espace de noms de ressource IPC.

- Pour les tâches qui utilisent le mode IPC `host`, les `systemControls` liés à l'espace de noms IPC ne sont pas pris en charge.
- Pour les tâches qui utilisent le mode IPC `task`, les `systemControls` liés à l'espace de noms IPC s'appliquent à tous les conteneurs au sein d'une tâche.

Mode PID

`pidMode`

Type : String

Valeurs valides : `host` | `task`

Obligatoire : non

Espace de noms de processus à utiliser pour les conteneurs de la tâche. Les valeurs valides sont `host` ou `task`. Par exemple, la surveillance des sidecars peut avoir besoin de `pidMode` pour accéder à des informations sur d'autres conteneurs exécutés dans le cadre de la même tâche.

Si `host` est spécifié, tous les conteneurs des tâches ayant spécifié le mode PID `host` sur la même instance de conteneur partagent le même espace de noms de processus avec l'instance Amazon EC2 hôte.

Si `task` est spécifié, tous les conteneurs de la tâche spécifiée partagent le même espace de noms de processus.

Si aucune valeur n'est spécifiée, la valeur par défaut est un espace de noms privé pour chaque conteneur.

Si le mode PID `host` est utilisé, il existe un risque accru d'exposition à un espace de noms de processus indésirable.

> **Note**
>
> Ce paramètre n'est pas pris en charge par les conteneurs Windows.

Injection de pannes

`enableFaultInjection`

Type : Boolean

Valeurs valides : `true` | `false`

Obligatoire : non

Si ce paramètre est défini sur `true`, dans la charge utile d'une tâche, Amazon ECS accepte les demandes d'injection de défauts provenant des conteneurs de la tâche. Par défaut, ce paramètre est défini sur `false`.

Modèle de définition de tâche Amazon ECS

Un modèle de définition de tâche vide est présenté ci-dessous. Vous pouvez utiliser ce modèle pour créer votre définition de tâche, qui peut ensuite être collée dans la zone de saisie JSON de la console

ou enregistrée dans un fichier et utilisée avec AWS CLI `--cli-input-json` option. Pour plus d'informations, consultez Paramètres de définition de tâche Amazon ECS pour Fargate.

Modèle EC2

```
{
  "family": "",
  "taskRoleArn": "",
  "executionRoleArn": "",
  "networkMode": "none",
  "containerDefinitions": [
    {
      "name": "",
      "image": "",
      "repositoryCredentials": {
        "credentialsParameter": ""
      },
      "cpu": 0,
      "memory": 0,
      "memoryReservation": 0,
      "links": [""],
      "portMappings": [
        {
          "containerPort": 0,
          "hostPort": 0,
          "protocol": "tcp"
        }
      ],
      "restartPolicy": {
        "enabled": true,
        "ignoredExitCodes": [0],
        "restartAttemptPeriod": 180
      },
      "essential": true,
      "entryPoint": [""],
      "command": [""],
      "environment": [
        {
          "name": "",
          "value": ""
        }
      ],
      "environmentFiles": [
        {
```

```
          "value": "",
          "type": "s3"
        }
      ],
      "mountPoints": [
        {
          "sourceVolume": "",
          "containerPath": "",
          "readOnly": true
        }
      ],
      "volumesFrom": [
        {
          "sourceContainer": "",
          "readOnly": true
        }
      ],
      "linuxParameters": {
        "capabilities": {
          "add": [""],
          "drop": [""]
        },
        "devices": [
          {
            "hostPath": "",
            "containerPath": "",
            "permissions": ["read"]
          }
        ],
        "initProcessEnabled": true,
        "sharedMemorySize": 0,
        "tmpfs": [
          {
            "containerPath": "",
            "size": 0,
            "mountOptions": [""]
          }
        ],
        "maxSwap": 0,
        "swappiness": 0
      },
      "secrets": [
        {
          "name": "",
```

```
          "valueFrom": ""
        }
      ],
      "dependsOn": [
        {
          "containerName": "",
          "condition": "COMPLETE"
        }
      ],
      "startTimeout": 0,
      "stopTimeout": 0,
      "hostname": "",
      "user": "",
      "workingDirectory": "",
      "disableNetworking": true,
      "privileged": true,
      "readonlyRootFilesystem": true,
      "dnsServers": [""],
      "dnsSearchDomains": [""],
      "extraHosts": [
        {
          "hostname": "",
          "ipAddress": ""
        }
      ],
      "dockerSecurityOptions": [""],
      "interactive": true,
      "pseudoTerminal": true,
      "dockerLabels": {
        "KeyName": ""
      },
      "ulimits": [
        {
          "name": "nofile",
          "softLimit": 0,
          "hardLimit": 0
        }
      ],
      "logConfiguration": {
        "logDriver": "splunk",
        "options": {
          "KeyName": ""
        },
        "secretOptions": [
```

```
          {
            "name": "",
            "valueFrom": ""
          }
        ]
      },
      "healthCheck": {
        "command": [""],
        "interval": 0,
        "timeout": 0,
        "retries": 0,
        "startPeriod": 0
      },
      "systemControls": [
        {
          "namespace": "",
          "value": ""
        }
      ],
      "resourceRequirements": [
        {
          "value": "",
          "type": "InferenceAccelerator"
        }
      ],
      "firelensConfiguration": {
        "type": "fluentbit",
        "options": {
          "KeyName": ""
        }
      }
    }
  ],
  "volumes": [
    {
      "name": "",
      "host": {
        "sourcePath": ""
      },
      "configuredAtLaunch": true,
      "dockerVolumeConfiguration": {
        "scope": "shared",
        "autoprovision": true,
        "driver": "",
```

```
        "driverOpts": {
          "KeyName": ""
        },
        "labels": {
          "KeyName": ""
        }
      },
      "efsVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "transitEncryption": "DISABLED",
        "transitEncryptionPort": 0,
        "authorizationConfig": {
          "accessPointId": "",
          "iam": "ENABLED"
        }
      },
      "fsxWindowsFileServerVolumeConfiguration": {
        "fileSystemId": "",
        "rootDirectory": "",
        "authorizationConfig": {
          "credentialsParameter": "",
          "domain": ""
        }
      }
    }
  ],
  "placementConstraints": [
    {
      "type": "memberOf",
      "expression": ""
    }
  ],
  "requiresCompatibilities": ["EC2"],
  "cpu": "",
  "memory": "",
  "tags": [
    {
      "key": "",
      "value": ""
    }
  ],
  "pidMode": "task",
  "ipcMode": "task",
```

```
  "proxyConfiguration": {
    "type": "APPMESH",
    "containerName": "",
    "properties": [
      {
        "name": "",
        "value": ""
      }
    ]
  },
  "inferenceAccelerators": [
    {
      "deviceName": "",
      "deviceType": ""
    }
  ],
  "ephemeralStorage": {
    "sizeInGiB": 0
  },
  "runtimePlatform": {
    "cpuArchitecture": "X86_64",
    "operatingSystemFamily": "WINDOWS_SERVER_20H2_CORE"
  }
}
```

Modèle Fargate

Important

Pour Fargate, vous devez inclure le paramètre `operatingSystemFamily` avec l'une des valeurs suivantes :

- `LINUX`
- `WINDOWS_SERVER_2019_FULL`
- `WINDOWS_SERVER_2019_CORE`
- `WINDOWS_SERVER_2022_FULL`
- `WINDOWS_SERVER_2022_CORE`

```
{
    "family": "",
    "runtimePlatform": {"operatingSystemFamily": ""},
    "taskRoleArn": "",
    "executionRoleArn": "",
    "networkMode": "awsvpc",
    "platformFamily": "",
    "containerDefinitions": [
        {
            "name": "",
            "image": "",
            "repositoryCredentials": {"credentialsParameter": ""},
            "cpu": 0,
            "memory": 0,
            "memoryReservation": 0,
            "links": [""],
            "portMappings": [
                {
                    "containerPort": 0,
                    "hostPort": 0,
                    "protocol": "tcp"
                }
            ],
            "essential": true,
            "entryPoint": [""],
            "command": [""],
            "environment": [
                {
                    "name": "",
                    "value": ""
                }
            ],
            "environmentFiles": [
                {
                    "value": "",
                    "type": "s3"
                }
            ],
            "mountPoints": [
                {
                    "sourceVolume": "",
                    "containerPath": "",
                    "readOnly": true
```

```
            }
        ],
        "volumesFrom": [
            {
                "sourceContainer": "",
                "readOnly": true
            }
        ],
        "linuxParameters": {
            "capabilities": {
                "add": [""],
                "drop": [""]
            },
            "devices": [
                {
                    "hostPath": "",
                    "containerPath": "",
                    "permissions": ["read"]
                }
            ],
            "initProcessEnabled": true,
            "sharedMemorySize": 0,
            "tmpfs": [
                {
                    "containerPath": "",
                    "size": 0,
                    "mountOptions": [""]
                }
            ],
            "maxSwap": 0,
            "swappiness": 0
        },
        "secrets": [
            {
                "name": "",
                "valueFrom": ""
            }
        ],
        "dependsOn": [
            {
                "containerName": "",
                "condition": "HEALTHY"
            }
        ],
```

```
            "startTimeout": 0,
            "stopTimeout": 0,
            "hostname": "",
            "user": "",
            "workingDirectory": "",
            "disableNetworking": true,
            "privileged": true,
            "readonlyRootFilesystem": true,
            "dnsServers": [""],
            "dnsSearchDomains": [""],
            "extraHosts": [
                {
                    "hostname": "",
                    "ipAddress": ""
                }
            ],
            "dockerSecurityOptions": [""],
            "interactive": true,
            "pseudoTerminal": true,
            "dockerLabels": {"KeyName": ""},
            "ulimits": [
                {
                    "name": "msgqueue",
                    "softLimit": 0,
                    "hardLimit": 0
                }
            ],
            "logConfiguration": {
                "logDriver": "awslogs",
                "options": {"KeyName": ""},
                "secretOptions": [
                    {
                        "name": "",
                        "valueFrom": ""
                    }
                ]
            },
            "healthCheck": {
                "command": [""],
                "interval": 0,
                "timeout": 0,
                "retries": 0,
                "startPeriod": 0
            },
```

```
            "systemControls": [
                {
                    "namespace": "",
                    "value": ""
                }
            ],
            "resourceRequirements": [
                {
                    "value": "",
                    "type": "GPU"
                }
            ],
            "firelensConfiguration": {
                "type": "fluentd",
                "options": {"KeyName": ""}
            }
        }
    ],
    "volumes": [
        {
            "name": "",
            "host": {"sourcePath": ""},
            "configuredAtLaunch":true,
            "dockerVolumeConfiguration": {
                "scope": "task",
                "autoprovision": true,
                "driver": "",
                "driverOpts": {"KeyName": ""},
                "labels": {"KeyName": ""}
            },
            "efsVolumeConfiguration": {
                "fileSystemId": "",
                "rootDirectory": "",
                "transitEncryption": "ENABLED",
                "transitEncryptionPort": 0,
                "authorizationConfig": {
                    "accessPointId": "",
                    "iam": "ENABLED"
                }
            }
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "cpu": "",
```

```
    "memory": "",
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "ephemeralStorage": {"sizeInGiB": 0},
    "pidMode": "task",
    "ipcMode": "none",
    "proxyConfiguration": {
        "type": "APPMESH",
        "containerName": "",
        "properties": [
            {
                "name": "",
                "value": ""
            }
        ]
    },
    "inferenceAccelerators": [
        {
            "deviceName": "",
            "deviceType": ""
        }
    ]
}
```

Vous pouvez générer ce modèle de définition de tâche à l'aide de la commande AWS CLI suivante.

```
aws ecs register-task-definition --generate-cli-skeleton
```

Exemple de définitions de tâches Amazon ECS

Vous pouvez copier les exemples et extraits de code pour commencer à créer vos propres définitions de tâches.

Vous pouvez copier les exemples, puis les coller lorsque vous utilisez l'option Configurer via JSON dans la console classique. Assurez-vous de personnaliser les exemples, en utilisant par exemple votre ID de compte. Vous pouvez inclure les extraits dans le JSON de définition de tâche. Pour plus

d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console et Paramètres de définition de tâche Amazon ECS pour Fargate.

Pour plus d'exemples de définition de tâches, voir AWS Exemples de définitions de tâches sur GitHub.

Rubriques

- Serveur Web
- splunk pilote de journal
- couramment pilote de journal
- gelf pilote de journal
- Charges de travail sur les instances externes
- Image Amazon ECR et rôle IAM de définition de tâche
- Point d'entrée avec commande
- Dépendances du conteneur
- Volumes dans les définitions de tâches
- Exemples de définitions de tâche Windows

Serveur Web

Voici un exemple de définition de tâche utilisant des conteneurs Linux sur Fargate qui configure un serveur Web :

```
{
   "containerDefinitions": [
      {
         "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\""
         ],
         "entryPoint": [
            "sh",
            "-c"
```

```
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

Voici un exemple de définition de tâche à l'aide des conteneurs Windows sur Fargate qui configure un serveur Web :

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<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>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

`splunk` pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `splunk` dans une définition de tâche qui envoie les journaux à un service distant. Le paramètre de jeton Splunk est spécifiée en tant qu'option secrète, car il peut être traité comme des données sensibles. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

```
"containerDefinitions": [{
  "logConfiguration": {
   "logDriver": "splunk",
   "options": {
    "splunk-url": "https://cloud.splunk.com:8080",
```

```
    "tag": "tag_name",
   },
   "secretOptions": [{
    "name": "splunk-token",
    "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:splunk-token-
KnrBkD"
}],
```

couramment pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `fluentd` dans une définition de tâche qui envoie les journaux à un service distant. La valeur `fluentd-address` est spécifiée en tant qu'option secrète, car elle peut être traitée comme des données sensibles. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "fluentd",
  "options": {
   "tag": "fluentd demo"
  },
  "secretOptions": [{
   "name": "fluentd-address",
   "valueFrom": "arn:aws:secretsmanager:region:aws_account_id:secret:fluentd-address-
KnrBkD"
  }]
 },
 "entryPoint": [],
 "portMappings": [{
            "hostPort": 80,
            "protocol": "tcp",
            "containerPort": 80
            },
            {
  "hostPort": 24224,
  "protocol": "tcp",
  "containerPort": 24224
 }]
}],
```

`gelf` pilote de journal

L'extrait suivant montre comment utiliser le pilote de journal `gelf` dans une définition de tâche qui envoie les journaux à un hôte distant exécutant Logstash qui accepte les journaux Gelf sous forme d'entrée. Pour plus d'informations, consultez `logConfiguration`.

```
"containerDefinitions": [{
 "logConfiguration": {
  "logDriver": "gelf",
  "options": {
   "gelf-address": "udp://logstash-service-address:5000",
   "tag": "gelf task demo"
  }
 },
 "entryPoint": [],
 "portMappings": [{
   "hostPort": 5000,
   "protocol": "udp",
   "containerPort": 5000
  },
  {
   "hostPort": 5000,
   "protocol": "tcp",
   "containerPort": 5000
  }
 ]
}],
```

Charges de travail sur les instances externes

Lors de l'enregistrement d'une définition de tâche Amazon ECS, utilisez le paramètre `requiresCompatibilities` et spécifiez `EXTERNAL`, qui valide la compatibilité de la définition de tâche lors de l'exécution d'applications Amazon ECS sur vos instances externes. Si vous utilisez la console pour enregistrer une définition de tâche, vous devez utiliser l'éditeur JSON. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

> **Important**
>
> Si vos tâches nécessitent un rôle IAM d'exécution de tâche, assurez-vous qu'il est spécifié dans la définition de tâche.

Lorsque vous déployez votre application, utilisez le type de lancement `EXTERNAL` lors de la création de votre service ou de l'exécution de votre tâche autonome.

Voici un exemple de définition de tâche.

Linux

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "nginx",
  "image": "public.ecr.aws/nginx/nginx:latest",
  "memory": 256,
  "cpu": 256,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
  }]
 }],
 "networkMode": "bridge",
 "family": "nginx"
}
```

Windows

```
{
 "requiresCompatibilities": [
  "EXTERNAL"
 ],
 "containerDefinitions": [{
  "name": "windows-container",
  "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-ltsc2019",
  "memory": 256,
  "cpu": 512,
  "essential": true,
  "portMappings": [{
   "containerPort": 80,
   "hostPort": 8080,
   "protocol": "tcp"
```

```
  }]
 }],
 "networkMode": "bridge",
 "family": "windows-container"
}
```

Image Amazon ECR et rôle IAM de définition de tâche

L'extrait suivant utilise une image Amazon ECR appelée `aws-nodejs-sample` avec la balise `v1` à partir du registre `123456789012.dkr.ecr.us-west-2.amazonaws.com`. Le conteneur de cette tâche hérite des autorisations IAM du rôle `arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole`. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "123456789012.dkr.ecr.us-west-2.amazonaws.com/aws-nodejs-
sample:v1",
            "memory": 200,
            "cpu": 10,
            "essential": true
        }
    ],
    "family": "example_task_3",
    "taskRoleArn": "arn:aws:iam::123456789012:role/AmazonECSTaskS3BucketRole"
}
```

Point d'entrée avec commande

L'extrait suivant illustre la syntaxe d'un conteneur Docker qui utilise un point d'entrée et un argument de commande. Ce conteneur enverra un ping à `example.com` quatre fois, puis il s'arrêtera.

```
{
    "containerDefinitions": [
        {
            "memory": 32,
            "essential": true,
            "entryPoint": ["ping"],
```

```
            "name": "alpine_ping",
            "readonlyRootFilesystem": true,
            "image": "alpine:3.4",
            "command": [
                "-c",
                "4",
                "example.com"
            ],
            "cpu": 16
        }
    ],
    "family": "example_task_2"
}
```

Dépendances du conteneur

Cet extrait illustre la syntaxe d'une définition de tâche avec plusieurs conteneurs où la dépendance de conteneur est spécifiée. Dans la définition de tâche suivante, le conteneur `envoy` doit atteindre un état sain, déterminé par les paramètres de surveillance de l'état du conteneur requis avant que le conteneur app démarre. Pour plus d'informations, consultez Dépendances du conteneur.

```
{
  "family": "appmesh-gateway",
  "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
  },
  "proxyConfiguration":{
      "type": "APPMESH",
      "containerName": "envoy",
      "properties": [
          {
              "name": "IgnoredUID",
              "value": "1337"
          },
          {
              "name": "ProxyIngressPort",
              "value": "15000"
          },
          {
              "name": "ProxyEgressPort",
              "value": "15001"
          },
          {
```

```
                "name": "AppPorts",
                "value": "9080"
            },
            {
                "name": "EgressIgnoredIPs",
                "value": "169.254.170.2,169.254.169.254"
            }
        ]
    },
    "containerDefinitions": [
      {
        "name": "app",
        "image": "application_image",
        "portMappings": [
          {
            "containerPort": 9080,
            "hostPort": 9080,
            "protocol": "tcp"
          }
        ],
        "essential": true,
        "dependsOn": [
          {
            "containerName": "envoy",
            "condition": "HEALTHY"
          }
        ]
      },
      {
        "name": "envoy",
        "image": "840364872350.dkr.ecr.region-code.amazonaws.com/aws-appmesh-
envoy:v1.15.1.0-prod",
        "essential": true,
        "environment": [
          {
            "name": "APPMESH_VIRTUAL_NODE_NAME",
            "value": "mesh/meshName/virtualNode/virtualNodeName"
          },
          {
            "name": "ENVOY_LOG_LEVEL",
            "value": "info"
          }
        ],
        "healthCheck": {
```

```
        "command": [
          "CMD-SHELL",
          "echo hello"
        ],
        "interval": 5,
        "timeout": 2,
        "retries": 3
      }
    }
  ],
  "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
  "networkMode": "awsvpc"
}
```

Volumes dans les définitions de tâches

Utilisez ce qui suit pour comprendre comment spécifier des volumes dans les tâches.

- Pour plus d'informations sur la configuration d'un volume Amazon EBS, consultez la section Spécification de la configuration du volume Amazon EBS lors du déploiement Amazon ECS.
- Pour plus d'informations sur la configuration d'un volume Amazon EFS, consultez la section Configuration des systèmes de fichiers Amazon EFS pour Amazon ECS à l'aide de la console.
- Pour plus d'informations sur la configuration d'un volume FSx for Windows File Server, consultez la section Configuration des systèmes de fichiers FSx for Windows File Server pour Amazon ECS.
- Pour plus d'informations sur la configuration d'un volume Docker, consultez la section Exemples de volumes Docker pour Amazon ECS.
- Pour plus d'informations sur la configuration d'un montage lié, consultez la section Exemples de montage lié pour Amazon ECS.

Exemples de définitions de tâche Windows

Voici un exemple de définition de tâche qui peut vous aider à commencer à utiliser les conteneurs Windows sur Amazon ECS.

Example Exemple d'application de console Amazon ECS pour Windows

La définition de tâche suivante est l'exemple d'application de console Amazon ECS produit dans l'assistant de première exécution pour Amazon ECS ; il a été transféré pour utiliser l'image de conteneur `microsoft/iis` Windows.

```
{
  "family": "windows-simple-iis",
  "containerDefinitions": [
    {
      "name": "windows_sample_app",
      "image": "mcr.microsoft.com/windows/servercore/iis",
      "cpu": 1024,
      "entryPoint":["powershell", "-Command"],
      "command":["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file -
Value '<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>'; C:\\ServiceMonitor.exe
 w3svc"],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80
        }
      ],
      "memory": 1024,
      "essential": true
    }
  ],
  "networkMode": "awsvpc",
  "memory": "1024",
  "cpu": "1024"
}
```

Clusters Amazon ECS

Un cluster Amazon ECS est un regroupement logique de tâches ou de services qui fournit la capacité d'infrastructure pour vos applications conteneurisées. Lorsque vous créez un cluster, vous avez le choix entre les trois principaux types d'infrastructure, optimisés pour différents cas d'utilisation et exigences opérationnelles.

Sélection du type de cluster approprié

Amazon ECS propose trois types d'infrastructure pour vos clusters. Choisissez le type qui correspond le mieux à vos exigences en matière de charge de travail, à vos préférences opérationnelles et à vos objectifs d'optimisation des coûts :

Instances gérées Amazon ECS (recommandées)

Idéal pour la plupart des charges de travail - AWS gère entièrement les instances Amazon EC2 sous-jacentes, y compris le provisionnement, l'application de correctifs et la mise à l'échelle. Cette option offre un équilibre optimal entre performances, rentabilité et simplicité opérationnelle.

À utiliser lorsque :

- Tu veux AWS pour gérer la gestion de l'infrastructure
- vous avez besoin d'une puissance de calcul rentable avec optimisation automatique ;
- vous souhaitez vous concentrer sur vos applications plutôt que sur l'infrastructure ;
- vous avez besoin de performances prévisibles avec une mise à l'échelle flexible.

Fargate

Calcul sans serveur : ne payez que pour les ressources utilisées par vos tâches, sans avoir à gérer aucune infrastructure. Idéal pour les charges de travail variables et pour démarrer rapidement.

À utiliser lorsque :

- vous voulez des opérations totalement sans serveur ;
- vous avez des charges de travail imprévisibles ou variables ;
- vous souhaitez minimiser les frais d'exploitation ;

- vous avez besoin d’un déploiement et d’une capacité de mise à l’échelle rapides.

Instances Amazon EC2

Contrôle total : vous gérez directement les instances Amazon EC2 sous-jacentes, y compris la sélection, la configuration et la maintenance des instances.

À utiliser lorsque :

- vous avez besoin de types d’instances ou de configurations spécifiques ;
- vous disposez d’une infrastructure Amazon EC2 existante à exploiter ;
- vous avez besoin d’AMI personnalisées ou de logiciels spécialisés ;
- vous avez besoin d’un contrôle maximal sur l’infrastructure sous-jacente.

> **Note**
>
> Les instances gérées Amazon ECS constituent le choix recommandé pour la plupart des nouvelles charges de travail, car elles offrent la meilleure combinaison de performances, d'optimisation des coûts et de simplicité opérationnelle tout en permettant AWS pour gérer les tâches de gestion de l'infrastructure.

Composants d’un cluster

Outre la capacité de l’infrastructure, un cluster est constitué des ressources suivantes :

- Réseau (VPC et sous-réseau) sur lequel s'exécutent vos tâches et services

  Lorsque vous utilisez des instances gérées Amazon ECS ou des instances Amazon EC2 pour la capacité, le sous-réseau peut se trouver dans des zones de disponibilité, des zones locales, des zones Wavelength ou AWS Outposts.
- Un espace de noms facultatif

  Un espace de noms est utilisé pour la communication de service à service avec Service Connect.
- Option de surveillance

  CloudWatch Container Insights est proposé à un coût supplémentaire et est un service entièrement géré. Il collecte, agrège et résume automatiquement les métriques et les journaux Amazon ECS.

Concepts de cluster

Vous trouverez ci-après des concepts généraux sur les clusters Amazon ECS.

- Vous créez des clusters pour séparer vos ressources.
- Les clusters sont Région AWS spécifique.
- Les clusters peuvent avoir l'un des états suivants.

  ACTIF

  Le cluster est prêt à accepter des tâches et, le cas échéant, vous pouvez enregistrer des instances de conteneur auprès du cluster.

  PROVISIONING

  Le cluster est associé à des fournisseurs de capacité et les ressources nécessaires au fournisseur de capacité sont en cours de création.

  DEPROVISIONING

  Le cluster est associé à des fournisseurs de capacité et les ressources nécessaires au fournisseur de capacité sont en cours de suppression.

  ÉCHEC

  Le cluster est associé à des fournisseurs de capacité et la création des ressources nécessaires au fournisseur de capacité a échoué.

  INACTIVE

  Le cluster a été supprimé. Les clusters ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Ce comportement est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des clusters `INACTIVE`.
- Un cluster peut contenir une combinaison de tâches qui sont hébergées sur des instances gérées Amazon ECS, AWS Fargate, des instances Amazon EC2 ou des instances externes. Les tâches peuvent être exécutées sur une infrastructure d'instances gérées Amazon ECS, Fargate ou EC2 en tant que type de lancement ou stratégie de fournisseur de capacité. Si vous utilisez des fournisseurs de capacité EC2, Amazon ECS ne surveille ni ne met à l'échelle la capacité des groupes Amazon EC2 Auto Scaling.
- Un cluster peut contenir un mélange de fournisseurs de capacité d'instances gérées Amazon ECS, de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité d'instances

gérées Amazon ECS, des fournisseurs de capacité du groupe Auto Scaling ou des fournisseurs de capacité Fargate.

- Vous pouvez utiliser différents types d'instances pour les instances gérées Amazon ECS et EC2, ou pour les fournisseurs de capacité du groupe Auto Scaling. Une instance ne peut être enregistrée que dans un seul cluster à la fois.
- Vous pouvez restreindre l'accès aux clusters en créant des politiques IAM personnalisées. Pour plus d'informations, consultez la section Exemples de cluster Amazon ECS dans Identity-based exemples de politiques pour Amazon Elastic Container Service.
- Vous pouvez utiliser Service Auto Scaling pour mettre à l'échelle les tâches Fargate. Pour plus d'informations, consultez Mise à l'échelle automatique de votre service Amazon ECS.
- Vous pouvez configurer un espace de noms Service Connect par défaut pour un cluster. Une fois que vous avez défini un espace de noms Service Connect par défaut, tous les nouveaux services créés dans le cluster peuvent être ajoutés en tant que services clients dans l'espace de noms en activant Service Connect. Aucune configuration supplémentaire n'est requise. Pour plus d'informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.

Fournisseurs de capacité

Les fournisseurs de capacité Amazon ECS gèrent la mise à l'échelle de l'infrastructure pour les tâches de vos clusters. Chaque cluster dispose d'un ou plusieurs fournisseurs de capacité et éventuellement d'une stratégie de fournisseur de capacité facultative. Vous pouvez attribuer une stratégie de fournisseur de capacité par défaut au cluster. La stratégie de fournisseur de capacité détermine la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster. Lorsque vous exécutez une tâche autonome ou que vous créez un service, vous pouvez utiliser soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut. La stratégie de fournisseur de capacité par défaut du cluster ne s'applique que lorsque vous ne spécifiez pas de type de lancement ou de stratégie de fournisseur de capacité pour votre tâche ou votre service. Si vous fournissez l'un de ces paramètres, la stratégie par défaut n'est pas utilisée.

Amazon ECS propose trois types de fournisseurs de capacité pour vos clusters :

Fournisseurs de capacité Amazon ECS Managed Instances

AWS gère entièrement les instances Amazon EC2 sous-jacentes, y compris le provisionnement, l'application de correctifs, la mise à l'échelle et la gestion du cycle de vie. Ce choix offre un

équilibre optimal entre performances, rentabilité et simplicité opérationnelle. Les fournisseurs de capacité d'instances gérées Amazon ECS optimisent automatiquement la sélection et la mise à l'échelle des instances en fonction de vos exigences en matière de charge de travail.

Avec les instances gérées Amazon ECS, vous bénéficiez des avantages suivants :

- Allocation et mise à l'échelle automatiques des instances
- Correctifs et mises à jour de sécurité gérés
- Optimisation des coûts grâce à la sélection intelligente des instances
- Frais d'exploitation réduits

Fournisseurs de capacité Fargate

Calcul sans serveur où vous ne payez que pour les ressources utilisées par vos tâches, sans aucune infrastructure à gérer. Il vous suffit d'associer les fournisseurs de capacité prédéfinis (Fargate et Fargate Spot) au cluster.

Fournisseurs de capacité de groupe Auto Scaling

Lorsque vous utilisez des instances Amazon EC2 pour votre capacité, vous utilisez un groupe Auto Scaling pour gérer les instances Amazon EC2. L'autoscaling permet de vous assurer que vous disposez du bon nombre d'instances Amazon EC2 disponibles pour gérer la charge de l'application. Vous avez un contrôle total sur l'infrastructure sous-jacente.

Un cluster peut contenir une combinaison de tâches qui sont hébergées sur des instances gérées Amazon ECS, AWS Fargate, des instances Amazon EC2 ou des instances externes. Les tâches peuvent être exécutées sur une infrastructure d'instances gérées Amazon ECS, Fargate ou EC2 en tant que type de lancement ou stratégie de fournisseur de capacité. Si vous utilisez EC2 comme type de lancement, Amazon ECS ne surveille ni ne met à l'échelle la capacité des groupes Amazon EC2 Auto Scaling.

Un cluster peut contenir un mélange de fournisseurs de capacité d'instances gérées Amazon ECS, de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité d'instances gérées Amazon ECS, des fournisseurs de capacité du groupe Auto Scaling ou des fournisseurs de capacité Fargate.

Clusters pour les instances gérées Amazon ECS

Les fournisseurs de capacité Amazon ECS gèrent la mise à l'échelle de l'infrastructure pour les tâches de vos clusters. Après avoir créé le cluster, vous créez un ou plusieurs fournisseurs de capacité et éventuellement une stratégie de fournisseur de capacité facultative pour le cluster. La stratégie de fournisseur de capacité détermine la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster. Lorsque vous exécutez une tâche autonome ou que vous créez un service, vous pouvez utiliser soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut.

Les instances gérées Amazon ECS dispose des fournisseurs de capacité suivants.

Type	Critères utilisés pour choisir les instances

Par défaut	Les instances les plus économiques qui répondent aux exigences suivantes en matière de définition des tâches et de paramètres de service :
	• Définition de tâche
	  • opérantSystemFamily
	  • cpuArchitecture
	  • cpu
	  • memory
	• Définition des services
	  • placementConstraints
	  • placementStrategy

Personnalisé	Les instances qui répondent aux exigences d'attribut et de type que vous spécifiez lors de la création du cluster. Pour plus d’informations sur les attributs, consultez la section Attributs d’instance de conteneur Amazon ECS. Pour plus d’informations sur les types d’instance, consultez la section Spécifications des types

Type	Critères utilisés pour choisir les instances
	d’instance Amazon EC2 dans Types d’instance Amazon EC2.

Amazon ECS lance les instances et les associe au fournisseurs de capacité d’instances gérées Amazon ECS. Pour le fournisseur de capacité personnalisé, Amazon ECS crée également le fournisseur de capacité.

Création d’un cluster pour les instances gérées Amazon ECS

Vous créez un cluster pour définir l’infrastructure sur laquelle vos tâches et vos services s’exécutent.

Lorsque vous créez un cluster pour les instances gérées Amazon ECS, Amazon ECS crée automatiquement un fournisseur de capacité d'instances gérées avec des configurations par défaut. Ce fournisseur de capacité sélectionne les types d'instances les plus rentables pour vos charges de travail. Vous pouvez également créer des fournisseurs de capacité personnalisés si vous avez besoin d’attributs ou de types d’instance spécifiques.

Pour simplifier au maximum le processus de création de clusters, la console propose des sélections par défaut pour de nombreux choix.

- Crée un espace de noms par défaut dans AWS Cloud Map qui porte le même nom que le cluster. Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire.

  Pour plus d’informations, consultez Interconnexion des services Amazon ECS.

Vous pouvez modifier les options suivantes :

- Modifiez l'espace de noms par défaut associé au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. L'espace de noms par défaut porte le même nom que le cluster. Pour plus d’informations, consultez Interconnexion des services Amazon ECS.

- Attribuer un AWS KMS clé pour votre stockage géré. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Ajoutez des balises pour vous aider à identifier vos clusters.

Conditions préalables

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour plus d'informations, consultez the section called "Exemples de cluster Amazon ECS".

L'utilisateur qui crée le cluster doit disposer d'une autorisation supplémentaire : `iam:CreateServiceLinkedRole`.

Lorsque vous ne spécifiez pas `instanceRequirements`, Amazon ECS sélectionne automatiquement les types d'instance les plus optimisés en termes de coûts en fonction de vos exigences en matière de définition des tâches. Pour utiliser des attributs ou des types d'instance spécifiques, créez un fournisseur de capacité avec `instanceRequirements`.

Découvrez comment choisir vos instances. Pour plus d'informations, consultez Pratiques exemplaires en matière de sélection d'instances pour les instances gérées Amazon ECS.

Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :

- Rôle de l'infrastructure - Permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

  Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
- Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

  Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.

Procédure pour la console

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :

   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, saisissez un nom unique dans Espace de nom.
6. Pour Fournisseur de capacité personnalisé, procédez comme suit :

   - Pour Sélectionner une méthode d’obtention de capacité EC2, choisissez Instances gérées Amazon ECS.
   - Pour Profil d’instance, choisissez le rôle du profil d’instance.
   - Pour Rôle d’infrastructure, choisissez le rôle d’infrastructure.
   - Pour utiliser un fournisseur de capacité personnalisé, pour Sélection de l’instance, sélectionnez Utiliser une instance personnalisée. Ensuite, pour chaque attribut, saisissez la valeur de l’attribut.
7. (Facultatif) Utilisez Container Insights, développez Surveillance, puis sélectionnez une des options suivantes :

   - Pour utiliser Container Insights avec observabilité améliorée comme recommandé, sélectionnez Container Insights avec observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
8. (Facultatif) Chiffrez les données sur le stockage géré. En dessous Chiffrement, pour Stockage géré, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de stockage gérées.
9. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Valeur, saisissez la valeur de clé.

10. Cliquez sur Créer.

AWS CLI procédure

Vous pouvez créer un cluster pour les instances gérées Amazon ECS à l'aide de l' AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS depuis AWS AWS CLI, des kits de développement logiciel et l'API Amazon ECS sur IPv4 et IPv6. Pour plus d'informations, consultez the section called "Utilisation des points de terminaison Dual-Stack".

Pour créer un nouveau cluster (AWS CLI)

1. Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name managed-instances-cluster
```

Sortie :

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "managed-instances-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
```

```
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/managed-instances-
cluster"
    }
}
```

2. (Facultatif) Pour activer Container Insights avec observabilité améliorée pour votre cluster, utilisez la commande suivante :

```
aws ecs put-account-setting --name containerInsights --value enhanced
```

3. (Facultatif) Pour ajouter des balises à votre cluster, utilisez la commande suivante :

```
aws ecs tag-resource --resource-arn
 arn:aws:ecs:region:aws_account_id:cluster/managed-instances-cluster --tags
 key=Environment,value=Production
```

Étapes suivantes

Créez une définition de tâche pour les instances gérées Amazon ECS. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Exécutez vos applications en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d'informations, consultez les ressources suivantes :

- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mise à jour propagée Amazon ECS

Mise à jour d'un cluster pour l'utilisation des instances gérées Amazon ECS

Vous pouvez mettre à jour un cluster existant pour utiliser les instances gérées Amazon ECS.

Lorsque vous ajoutez des instances gérées Amazon ECS à votre cluster, Amazon ECS crée automatiquement un fournisseur de capacité d'instances gérées avec des configurations par défaut. Ce fournisseur de capacité sélectionne les types d'instances à usage général les plus optimisés en termes de coûts pour vos charges de travail. Vous pouvez également créer des fournisseurs de capacité personnalisés si vous avez besoin d'attributs ou de types d'instance spécifiques.

Conditions préalables

Lorsque vous ne spécifiez pas `instanceRequirements`, Amazon ECS sélectionne automatiquement les types d'instance les plus optimisés en termes de coûts en fonction de vos exigences en matière de définition des tâches. Pour utiliser des attributs ou des types d'instance spécifiques, créez un fournisseur de capacité avec `instanceRequirements`.

Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :

- Rôle de l'infrastructure - Permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

  Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
- Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

  Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.

Considérations relatives à la mise à jour

Tenez compte des éléments suivants lorsque vous mettez à jour un cluster pour instances gérées Amazon ECS :

- Tâches en cours : la mise à jour des paramètres du cluster n'affecte pas les tâches en cours d'exécution. Les modifications s'appliqueront aux nouvelles tâches lancées après la mise à jour.
- Changements de fournisseur de capacité : si vous modifiez les paramètres du fournisseur de capacité, les instances gérées existantes continueront de fonctionner, mais les nouvelles instances utiliseront la configuration mise à jour.
- Surveillance des modifications : l'activation ou la désactivation de Container Insights affectera la collecte de métriques pour l'ensemble du cluster.

Procédure pour la console

Pour mettre à jour un cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.

3. Dans le panneau de navigation, choisissez Clusters.
4. Sélectionnez Clusters et choisissez le cluster que vous souhaitez mettre à jour.
5. Choisissez Mettre à jour le cluster.
6. (Facultatif) Pour modifier les paramètres du fournisseur de capacité, sous Fournisseur de capacité personnalisé, mettez à jour les éléments suivants selon les besoins :
   - Pour Profil d'instance, choisissez un rôle de profil d'instance différent si nécessaire.
   - Pour Rôle d'infrastructure, choisissez un rôle d'infrastructure différent si nécessaire.
   - Pour utiliser un fournisseur de capacité personnalisé, pour Sélection de l'instance, mettez à jour les paramètres de Valeur d'attribut.
7. Choisissez Mettre à jour.

AWS CLI procédure

Vous pouvez mettre à jour un cluster pour les instances gérées Amazon ECS à l'aide de l' AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS depuis AWS AWS CLI, des kits de développement logiciel et l'API Amazon ECS sur IPv4 et IPv6. Pour plus d'informations, consultez the section called "Utilisation des points de terminaison Dual-Stack".

Pour mettre à jour un cluster (AWS CLI)

1. Créer un fournisseur de capacité pour . Exécutez la commande suivante :

   Remplacez le *user-input* avec tes valeurs.

```
aws ecs create-capacity-provider \
    --name my-managed-instances-provider \
    --managed-instances-provider \
    --instance-profile arn:aws:iam::123456789012:instance-profile/
ecsInstanceProfile \
```

```
    --infrastructure-role-arn arn:aws:iam::123456789012:role/ecsInfrastructureRole
\
    --instance-requirements '{
        "vCpuCount": {"min": 2, "max": 8},
        "memoryMiB": {"min": 4096, "max": 16384}
    }
```

2. Ajoutez le fournisseur de capacité au cluster à l'aide de la commande suivante :

   Remplacez le *user-input* avec tes valeurs.

```
aws ecs put-cluster-capacity-providers --cluster managed-instances-cluster --
capacity-providers my-managed-instances-provider --default-capacity-provider-
strategy capacityProvider=my-managed-instances-provider,weight=1
```

Fournisseurs de capacité d'instances gérées Amazon ECS

Les fournisseurs de capacité Amazon ECS Managed Instances fournissent un modèle de calcul en conteneur qui vous donne accès à la gamme complète de AWS fonctionnalités et offres Amazon EC2 tandis que AWS gère les responsabilités opérationnelles et de sécurité. AWS gère l'application de correctifs aux logiciels et aux systèmes d'exploitation, la mise à l'échelle des instances et la maintenance, vous offrant ainsi les avantages opérationnels de Fargate tout en conservant l'accès à tous AWS fonctionnalités et intégrations.

Amazon ECS crée des modèles de lancement pour vos instances gérées Amazon ECS. Cela définit la manière dont Amazon ECS lance les instances gérées Amazon ECS, y compris le profil d'instance pour vos tâches, la configuration du réseau et du stockage, les options de capacité et les exigences d'instance pour une sélection flexible du type d'instance.

Quand utiliser des fournisseurs de capacité personnalisés

Envisagez l'utilisation de fournisseurs de capacité personnalisés lorsque vos charges de travail nécessitent :

- Exigences de calcul spécifiques : applications nécessitant un calcul accéléré, des jeux d'instructions spécifiques au processeur, des performances réseau élevées ou des configurations de mémoire volumineuses qui ne sont pas disponibles avec les options Fargate standard.

- Charges de travail GPU : inférence d'apprentissage automatique, rendu d'image en temps réel, encodage vidéo, etc. GPU-accelerated applications nécessitant un accès aux GPU NVIDIA ou AMD.
- Réservations de capacité : Mission-critical charges de travail nécessitant une disponibilité prévisible des capacités.
- Observabilité avancée : outils de sécurité et de surveillance nécessitant un accès privilégié au système d'exploitation sous-jacent, tels que eBPF-based des solutions de surveillance ou des outils d'analyse de réseau.
- Optimisation des coûts : charges de travail qui peuvent bénéficier d'un placement multitâche, de composants d'infrastructure partagés ou qui doivent optimiser l'utilisation de types d'instances plus importants.

Options de surveillance

Les instances gérées Amazon ECS fournissent des fonctionnalités de surveillance complètes pour vous aider à suivre les performances, l'état et l'utilisation des ressources de vos charges de travail conteneurisées. Vous pouvez choisir parmi différents niveaux de surveillance en fonction de vos exigences opérationnelles.

- Surveillance de base : fournit des métriques essentielles à intervalles de cinq minutes pour la plupart des métriques et à intervalles d'une minute pour les vérifications de statut. Cette fonctionnalité est activée par défaut et n'entraîne aucuns frais supplémentaires.
- Surveillance détaillée : offre une visibilité améliorée avec toutes les métriques disponibles à intervalles d'une minute, permettant une détection et une réponse plus rapides aux problèmes opérationnels. Pour plus d'informations, consultez Surveillance détaillée des instances gérées Amazon ECS.

Les deux options de surveillance s'intègrent parfaitement à CloudWatch pour fournir des tableaux de bord, des alarmes et des réponses automatisées afin de maintenir des performances et une disponibilité optimales des applications.

Considérations relatives aux fournisseurs de capacité

Un cluster peut contenir un mélange de fournisseurs de capacité d'instances gérées Amazon ECS, de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité d'instances

gérées Amazon ECS, des fournisseurs de capacité du groupe Auto Scaling ou des fournisseurs de capacité Fargate.

Propagation de balises

Les fournisseurs de capacité pour les instances gérées Amazon ECS prennent en charge la propagation des balises. Avec la propagation des balises, toutes les ressources gérées par le fournisseur de capacité (l'instance gérée, l'instance de conteneur Amazon ECS, le modèle de lancement, les volumes, les interfaces réseau Elastic) sont étiquetées avec les mêmes balises spécifiées au niveau du fournisseur de capacité. Vous pouvez spécifier des balises lors de la création du fournisseur de capacité et activer la propagation des balises en spécifiant le paramètre `propagateTags` en tant que `CAPACITY_PROVIDER`.

Pour plus d'informations sur le balisage des instances gérées Amazon ECS, consultez la section Balises pour les instances gérées Amazon ECS.

Pratiques exemplaires pour la mise à jour des fournisseurs de capacité pour les instances gérées Amazon ECS

Pour bénéficier du plus haut niveau de sécurité et d'assistance en cas de restauration, nous recommandons de traiter les fournisseurs de capacités comme des ressources immuables. Lorsque vous devez mettre à jour la configuration d'un fournisseur de capacité, suivez ce flux de travail recommandé :

1. Créez un fournisseur de capacité avec votre configuration mise à jour au lieu de modifier le fournisseur existant.
2. Mettez à jour chaque service pour utiliser le nouveau fournisseur de capacité et permettre aux déploiements de se terminer.
3. Supprimez l'ancien fournisseur de capacité après avoir confirmé que la nouvelle configuration fonctionne comme prévu.

Cette approche offre plusieurs avantages :

- Déploiement contrôlé : vous pouvez mettre à jour les services un par un et en surveiller l'impact.
- Restauration facile : en cas de problème, vous pouvez rapidement rétablir les services pour utiliser le fournisseur de capacité précédent.
- Rayon d'action réduit : les problèmes liés à la nouvelle configuration n'affectent pas immédiatement toutes les charges de travail.

> **Note**
>
> Si vous utilisez CloudFormation, pensez à conserver l'ancien fournisseur de capacité jusqu'à un déploiement ultérieur afin de conserver la possibilité d'annuler les modifications apportées à votre stack.

Bien que vous puissiez mettre à jour les fournisseurs de capacité en place, cette approche crée un rayon d'explosion non contrôlé plus grand. In-place les mises à jour appliquent de nouveaux paramètres à toutes les nouvelles capacités provisionnées à l'avenir, mais ne déclenchent pas de déploiements de services. Cela signifie que vous ne découvrirez peut-être les problèmes de configuration que bien plus tard, lorsque vos services devront être mis à l'échelle.

Création d'un fournisseur de capacité pour les instances gérées Amazon ECS

Les instances gérées Amazon ECS font appel à des fournisseurs de capacité pour gérer la capacité de calcul de vos charges de travail. Lorsque vous créez un fournisseur de capacité sans spécifier `instanceRequirements`, Amazon ECS sélectionne automatiquement les solutions les plus optimisées en termes de coûts types d'instances à usage général. Vous pouvez créer des fournisseurs de capacité avec `instanceRequirements` pour spécifier les attributs d'instance tels que les types d'instance, les fabricants de processeurs, les types d'accélérateurs et d'autres exigences.

Les fournisseurs de capacité personnalisés utilisent la sélection du type d'instance basée sur des attributs, qui vous permet d'exprimer les exigences d'instance sous la forme d'un ensemble d'attributs. Ces exigences sont automatiquement transposées à tous les types d'instances Amazon EC2 correspondants, ce qui simplifie la création et la maintenance des configurations de types d'instances. Pour en savoir plus sur les exigences relatives aux instances et la sélection basée sur les attributs, consultez la documentation relative à la sélection du type d'instance basée sur les attributs de flotte d'Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Conditions préalables

Avant de commencer, assurez-vous d'avoir effectué les opérations suivantes :

- Déterminer le type de surveillance à utiliser. Pour plus d'informations, consultez the section called "Surveillance détaillée des instances gérées Amazon ECS".
- Avoir un cluster existant ou prévoir d'en créer un. Pour plus d'informations, consultez Création d'un cluster pour les instances gérées Amazon ECS.

- Vous disposez des rôles IAM requis pour les instances gérées Amazon ECS. Cela inclut notamment les éléments suivants :
  - Rôle de l'infrastructure - Permet à Amazon ECS de passer des appels vers AWS services en votre nom pour gérer l'infrastructure Amazon ECS Managed Instances.

    Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.
  - Profil d'instance : fournit des autorisations pour l'agent de conteneur Amazon ECS et le démon Docker exécutés sur des instances gérées.

    Pour plus d'informations, consultez Profil d'instance des instances gérées Amazon ECS.

Découvrez comment choisir vos instances. Pour plus d'informations, consultez Pratiques exemplaires en matière de sélection d'instances pour les instances gérées Amazon ECS.

Procédure pour la console

Pour créer un fournisseur de capacité pour instances gérées Amazon ECS (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez le nom de votre cluster.
5. Sur la page Clusters, choisissez l'onglet Infrastructure.
6. Dans la section Fournisseur de capacité, choisissez Créer un fournisseur de capacité.
7. Sous Configuration du fournisseur de capacité, configurez les éléments suivants :
   - Pour Nom du fournisseur de capacité, saisissez un nom unique pour votre fournisseur de capacité.
   - Pour Type de fournisseur de capacité, choisissez Instances gérées Amazon ECS.
8. Sous Configuration de l'instance, configurez les éléments suivants :
   - Pour Profil d'instance, choisissez le rôle de profil d'instance créé pour les instances gérées Amazon ECS.
   - Pour Rôle d'infrastructure, choisissez le rôle d'infrastructure créé pour les instances gérées Amazon ECS.

9. Sous Exigences de l'instance, spécifiez les attributs de vos instances. Vous pouvez configurer n'importe quelle combinaison de ce qui suit :

   - Nombre de vCPU : spécifiez le nombre de vCPU (par exemple, `4` ou `8-16` pour une plage).
   - Mémoire (Mio) : spécifiez la quantité de mémoire en Mio (par exemple, `8192` ou `16384-32768` pour une plage).
   - Types d'instances : spécifiez des types d'instances spécifiques (par exemple, `m5.large,m5.xlarge,c5.large`).
   - Fabricants d'UC : choisissez parmi `intel`, `amd`, ou `amazon-web-services`.
   - Types d'accélérateurs : spécifiez les types d'accélérateurs tels que `gpu`, `fpga`, ou `inference`.
   - Nombre d'accélérateurs : spécifiez le nombre d'accélérateurs (par exemple, `1` ou `2-4` pour une plage).

10. Sous Configuration avancée, choisissez l'une des options de surveillance suivantes :

    - Avoir CloudWatch envoyez des mesures de vérification de statut, choisissez Basique.
    - Avoir CloudWatch envoyez toutes les métriques, choisissez Détaillé.

11. (Facultatif) Pour vous aider à identifier votre cluster, développez Balises, puis configurez vos balises.

    Pour activer la propagation des balises du fournisseur de capacité vers les ressources gérées, telles que les instances lancées depuis le fournisseur de capacité, pour Propager les balises depuis, choisissez Fournisseur de capacité.

    [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

    - Pour Key (Clé), saisissez le nom de la clé.
    - Pour Valeur, saisissez la valeur de clé.

12. Cliquez sur Créer.

AWS CLI procédure

Vous pouvez créer un fournisseur de capacité pour les instances gérées Amazon ECS à l'aide de l' AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

Pour créer un fournisseur de capacité pour les instances gérées Amazon ECS (AWS CLI)

1. Exécutez la commande suivante :

```
aws ecs create-capacity-provider --cli-input-json file://capacity-provider-
definition.json
```

Le fichier `capacity-provider-definition.json` suivant peut être utilisé pour spécifier les exigences de base de l'instance, la taille de stockage de l'instance et activer la propagation des balises :

```
{
    "name": "my-managed-instances-provider",
    "cluster": "my-cluster",
    "tags": [
        {
            "key": "version",
            "value": "test"
        }
    ],
    "managedInstancesProvider": {
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/
ecsInstanceRole",
            "instanceRequirements": {
                "vCpuCount": {
                    "min": 4,
                    "max": 8
                },
                "memoryMiB": {
                    "min": 8192,
                    "max": 16384
                }
            },
            "networkConfiguration": {
                "subnets": [
                    "subnet-abcdef01234567",
                    "subnet-bcdefa98765432"
                ],
                "securityGroups": [
```

```
                    "sg-0123456789abcdef"
                ]
            },
            "storageConfiguration": {
                "storageSizeGiB": 100
            },
            "monitoring": "basic"
        },
        "propagateTags": "CAPACITY_PROVIDER"
    }
}
```

2. Vérifiez que votre fournisseur de capacité a été créé avec succès :

```
aws ecs describe-capacity-providers \
    --capacity-providers my-managed-instances-provider
```

Étapes suivantes

Après avoir créé votre fournisseur de capacité, vous pouvez l'utiliser pour créer des services ou exécuter des tâches :

- Pour utiliser le fournisseur de capacité avec un service, consultez la section Création d'un déploiement de mise à jour propagée Amazon ECS.
- Pour utiliser le fournisseur de capacité avec des tâches autonomes, consultez la section Exécution d'une application en tant que tâche Amazon ECS.

Mise à jour de la surveillance des instances gérées Amazon ECS

Vous pouvez modifier l'option de surveillance pour votre fournisseur de capacité d'instances gérées Amazon ECS afin de passer d'une surveillance de base à une surveillance détaillée. Cela vous permet d'ajuster le niveau des données de surveillance collectées sans avoir à recréer le fournisseur de capacité.

Pour plus d'informations sur les options de surveillance, voir Surveillance des instances gérées Amazon ECS.

Procédure pour la console

Pour mettre à jour la surveillance des instances gérées Amazon ECS (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez le nom de votre cluster.
5. Sur la page Clusters, choisissez l'onglet Infrastructure.
6. Sous Configuration avancée, choisissez l'une des options de surveillance suivantes :
   - Avoir CloudWatch envoyez des mesures de vérification de statut, choisissez Basique.
   - Avoir CloudWatch envoyez toutes les métriques, choisissez Détaillé.
7. Choisissez Mettre à jour.

Pour mettre à jour les balises associées à un fournisseur de capacité d'instances gérées Amazon ECS existant, procédez comme suit :

1. Dans le panneau de navigation, choisissez Clusters.
2. Sur la page Clusters, choisissez Infrastructure.
3. Sur la page Infrastructure, choisissez le fournisseur de capacité que vous avez créé.
4. Sur la page Fournisseur de capacité, choisissez Balises.
5. Sous Balises, choisissez Gérer les balises.
6. Pour ajouter une balise, saisissez la clé et la valeur de la balise que vous voulez ajouter, puis choisissez Enregistrer. Pour ajouter plusieurs balises à la fois, choisissez Ajouter une balise pour chaque balise que vous voulez ajouter. Vous pouvez ajouter un maximum de 50 balises.

   Pour supprimer une identification, choisissez Supprimer.

   > **Note**
   >
   > Si la propagation des balises est activée, les balises ajoutées ou supprimées après la création du fournisseur de capacité ne s'appliquent pas aux ressources créées précédemment par le fournisseur de capacité.

AWS CLI procédure

Vous pouvez mettre à jour un fournisseur de capacité pour les instances gérées Amazon ECS à l'aide de l' AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

Pour mettre à jour la surveillance des instances gérées Amazon ECS (AWS CLI)

1. Pour activer la surveillance détaillée, utilisez la commande suivante :

```
aws ecs update-capacity-provider \
    --name my-managed-instances-provider \
    --managed-instances-provider '{
        "instanceLaunchTemplateUpdate": {
            "monitoring": "DETAILED"
        }
    }'
```

2. Pour activer la surveillance de base, utilisez la commande suivante :

```
aws ecs update-capacity-provider \
    --name my-managed-instances-provider \
    --managed-instances-provider '{
        "instanceLaunchTemplateUpdate": {
            "monitoring": "BASIC"
        }
    }'
```

Suppression d'un fournisseur de capacité d'instances gérées Amazon ECS

Si vous n'utilisez plus un fournisseur de capacité d'instances gérées Amazon ECS, vous pouvez le supprimer. Une fois le groupe supprimé, le fournisseur de capacité d'instances gérées Amazon ECS passe à l'état `INACTIVE`. Les fournisseurs de capacité ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des fournisseurs de capacité dont l'état est `INACTIVE`. Avant de supprimer le fournisseur de capacité d'instance gérée Amazon ECS, vous devez le supprimer de la stratégie de fournisseur de capacité de tous les services. Vous pouvez utiliser l'API `UpdateService` ou le flux de service de mise à jour dans la console Amazon ECS pour supprimer un fournisseur de capacité de la stratégie de fournisseur de

capacité d'un service. Utilisez l'option Forcer le nouveau déploiement pour vous assurer que toutes les tâches utilisant la capacité fournie par le fournisseur de capacité d'instances gérées Amazon ECS sont transférées vers les autres fournisseurs de capacité afin d'utiliser leur capacité.

Pour supprimer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures, le fournisseur de capacité Amazon ECS Managed Instances, puis choisissez Supprimer.
5. Dans la case de confirmation, entrez supprimer ***Amazon ECS Managed Instances capacity provider name***
6. Sélectionnez Supprimer.

Arrêt sécurisé des charges de travail Amazon ECS exécutées sur des instances gérées Amazon ECS

Vous pouvez utiliser la protection évolutive des tâches Amazon ECS pour empêcher que vos tâches ne soient interrompues par des événements de réduction horizontale liés à la réduction horizontale des services ou aux déploiements.

Certaines applications nécessitent un mécanisme permettant de protéger les tâches critiques contre toute interruption due à des événements de mise à l'échelle horizontale en période de faible utilisation ou lors de déploiements de services. Par exemple :

- Vous disposez d'une application asynchrone de traitement des files d'attente, telle qu'une tâche de transcodage vidéo, dans laquelle certaines tâches doivent s'exécuter pendant des heures, même lorsque l'utilisation cumulée des services est faible.
- Vous disposez d'une application de jeu exécutant des serveurs de jeu en tant que tâches, lesquels doivent continuer à fonctionner même si tous les utilisateurs se sont déconnectés afin de réduire la latence de démarrage lors d'un redémarrage du serveur.
- Lorsque vous déployez une nouvelle version de code, vous avez besoin que des tâches continuent à s'exécuter, car leur retraitement serait coûteux.

Pour empêcher les tâches appartenant à votre service de se terminer lors d'un événement de mise à l'échelle horizontale, définissez l'attribut `ProtectionEnabled` sur `true`. Lorsque vous définissez `ProtectionEnabled` sur true, les tâches sont protégées pendant deux heures par défaut. Vous pouvez ensuite personnaliser la période de protection à l'aide de l'attribut `ExpiresInMinutes`. Vous pouvez protéger vos tâches pendant au moins une minute et jusqu'à un maximum de 2 880 minutes (48 heures). Si vous utilisez AWS CLI, vous pouvez spécifier `--protection-enabled` option.

Une fois qu'une tâche a terminé son travail requis, vous pouvez définir l'attribut `ProtectionEnabled` sur `false`, ce qui permet à la tâche d'être interrompue par des événements ultérieurs de mise à l'échelle horizontale. Si vous utilisez AWS CLI, vous pouvez spécifier `--no-protection-enabled` option.

Mécanismes de protection évolutive des tâches

Vous pouvez définir et obtenir une protection évolutive des tâches à l'aide du point de terminaison de l'agent de conteneur Amazon ECS ou de l'API Amazon ECS.

- Point de terminaison de l'agent de conteneur Amazon ECS

  Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection. Utilisez cette approche pour les charges de travail basées sur les files d'attente ou le traitement des tâches.

  Lorsqu'un conteneur commence à traiter un travail, par exemple en consommant un message SQS, vous pouvez définir l'attribut `ProtectionEnabled` via le chemin `$ECS_AGENT_URI/task-protection/v1/state` du point de terminaison de protection évolutive des tâches depuis le conteneur. Amazon ECS n'interrompra pas cette tâche lors d'événements de mise à l'échelle horizontale. Une fois que votre tâche a terminé son travail, vous pouvez effacer l'attribut `ProtectionEnabled` en utilisant le même point de terminaison, ce qui rend la tâche susceptible d'être interrompue lors d'événements ultérieurs de mise à l'échelle horizontale.

  Pour de plus amples informations sur l'utilisation de point de terminaison de l'agent de conteneur Amazon ECS, consultez la section Point de terminaison de protection évolutive des tâches Amazon ECS.
- API Amazon ECS

  Vous pouvez utiliser l'API Amazon ECS pour définir et récupérer la protection évolutive des tâches si votre application dispose d'un composant qui suit l'état des tâches actives. Utilisez

`UpdateTaskProtection` pour marquer une ou plusieurs tâches comme protégées. Utilisez `GetTaskProtection` pour récupérer l'état de protection.

Un exemple de cette approche serait si votre application héberge des sessions de serveur de jeu comme tâches Amazon ECS. Lorsqu'un utilisateur se connecte à une session sur le serveur (tâche), vous pouvez marquer la tâche comme protégée. Une fois que l'utilisateur se déconnecte, vous pouvez soit retirer la protection spécifique à cette tâche, soit retirer périodiquement la protection pour des tâches similaires qui ne comportent plus de sessions actives, en fonction de vos besoins en termes de serveurs inactifs.

Pour plus d'informations, voir UpdateTaskProtection et GetTaskProtection dans le Référence de l'API Amazon Elastic Container Service.

Vous pouvez combiner les deux approches. Par exemple, utilisez le point de terminaison de l'agent Amazon ECS pour définir la protection des tâches depuis un conteneur et utilisez l'API Amazon ECS pour retirer la protection des tâches depuis votre service de contrôleur externe.

Considérations

Prenez en compte les points suivants avant d'utiliser la protection évolutive des tâches :

- La protection évolutive des tâches n'est prise en charge que pour les tâches déployées à partir d'un service.
- La protection évolutive des tâches est prise en charge par les tâches déployées à partir d'un service exécuté sur les instances gérées Amazon ECS.
- Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS, car l'agent Amazon ECS possède des mécanismes de nouvelle tentative intégrés et une interface plus simple.
- Vous pouvez réinitialiser la période d'expiration de la protection contre la mise à l'échelle horizontale des tâches en appelant `UpdateTaskProtection` pour une tâche pour laquelle la protection est déjà activée.
- Déterminez le temps nécessaire à une tâche pour effectuer le travail requis et définissez la propriété `expiresInMinutes` en conséquence. Si vous fixez une période d'expiration de la protection plus longue que nécessaire, vous devrez supporter des coûts et serez confronté à des retards dans le déploiement de nouvelles tâches.
- Considérations relatives au déploiement :

- Si le service utilise une mise à jour propagée, de nouvelles tâches seront créées, mais les tâches exécutant une ancienne version ne seront pas terminées avant la désactivation ou l'expiration de `protectionEnabled`. Vous pouvez ajuster le paramètre `maximumPercentage` dans la configuration du déploiement sur une valeur qui permet de créer de nouvelles tâches lorsque les anciennes tâches sont protégées.
- Si vous utilisez CloudFormation, la pile de mise à jour a un délai d'expiration de 3 heures. Par conséquent, si vous configurez la protection de vos tâches pendant plus de 3 heures, CloudFormation le déploiement peut entraîner un échec et un retour en arrière.

  Pendant la durée de protection de vos anciennes tâches, le CloudFormation expositions empilées `UPDATE_IN_PROGRESS`. Si la protection contre la mise à l'échelle horizontale des tâches est supprimée ou expire dans un délai de trois heures, votre déploiement réussira et passera au statut `UPDATE_COMPLETE`. Si le déploiement est bloqué dans l'état `UPDATE_IN_PROGRESS` pendant plus de 3 heures, il échouera et affichera un état `UPDATE_FAILED`, puis reviendra au dernier ensemble de tâches.
- Amazon ECS envoie des événements de service lorsque des tâches protégées permettent de maintenir un déploiement (continu ou blue/green) d'atteindre l'état stable, afin que vous puissiez prendre des mesures correctives. Lorsque vous essayez de mettre à jour l'état de protection d'une tâche et que vous recevez un message d'erreur `DEPLOYMENT_BLOCKED`, cela signifie que le service possède un nombre de tâches protégées supérieur au nombre de tâches souhaité pour le service. Pour résoudre cette erreur, effectuez l'une des opérations suivantes :
  - Attendez que la protection des tâches en cours expire. Définissez ensuite la protection des tâches.
  - Déterminez quelles tâches peuvent être interrompues. Utilisez ensuite `UpdateTaskProtection` avec l'option `protectionEnabled` définie sur `false` pour ces tâches.
  - Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

Autorisations IAM requises pour la protection évolutive des tâches

La tâche doit avoir le rôle de tâche Amazon ECS avec les autorisations suivantes :

- `ecs:GetTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `GetTaskProtection`.

- `ecs:UpdateTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `UpdateTaskProtection`.

Point de terminaison de protection évolutive des tâches Amazon ECS

L'agent de conteneur Amazon ECS injecte automatiquement la variable d'environnement `ECS_AGENT_URI` dans les conteneurs des tâches Amazon ECS afin de fournir une méthode permettant d'interagir avec le point de terminaison de l'API de l'agent de conteneur.

Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection.

Lorsqu'un conteneur commence à traiter une tâche, vous pouvez définir l'attribut `protectionEnabled` à l'aide du chemin d'accès `$ECS_AGENT_URI/task-protection/v1/state` du point de terminaison de protection évolutive des tâches à partir du conteneur.

Utilisez une requête PUT vers cette URI à partir d'un conteneur pour définir la protection évolutive des tâches. Une requête GET vers cette URI renvoie le statut de protection actuel d'une tâche.

Paramètres de requête de protection évolutive des tâches

Vous pouvez définir la protection évolutive des tâches à l'aide du point de terminaison `${ECS_AGENT_URI}/task-protection/v1/state` avec les paramètres de demande suivants.

`ProtectionEnabled`

Spécifiez `true` pour marquer une tâche à protéger. Spécifiez `false` si vous souhaitez supprimer la protection et rendre la tâche éligible à l'arrêt.

Type : Boolean

Obligatoire : oui

`ExpiresInMinutes`

Le nombre de minutes pendant lesquelles la tâche est protégée. Vous pouvez spécifier un minimum de 1 minute et aller jusqu'à 2 880 minutes (48 heures). Pendant cette période, votre tâche ne sera pas interrompue par des événements de mise à l'échelle horizontale provenant de l'autoscaling du service ou de déploiements. Une fois cette période écoulée, le paramètre `protectionEnabled` est défini sur `false`.

Si vous ne spécifiez pas de durée, la tâche est automatiquement protégée pendant 120 minutes (2 heures).

Type : Integer

Obligatoire : non

Les exemples suivants montrent comment définir la protection des tâches avec des durées différentes.

Exemple de protection d'une tâche avec la durée par défaut

Cet exemple montre comment protéger une tâche dont la durée par défaut est de 2 heures.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Exemple de protection d'une tâche pendant 60 minutes

Cet exemple montre comment protéger une tâche pendant 60 minutes à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Exemple de protection d'une tâche pendant 24 heures

Cet exemple montre comment protéger une tâche pendant 24 heures à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

La requête PUT renvoie la réponse suivante.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
```

```
}
```

Paramètres de réponse de protection évolutive des tâches

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de protection contre la mise à l'échelle horizontale des tâches `${ECS_AGENT_URI}/task-protection/v1/state`.

`ExpirationDate`

Période pendant laquelle la protection de la tâche expirera. Si la tâche n’est pas protégée, cette valeur est nulle.

`ProtectionEnabled`

État de protection de la tâche. Si la protection évolutive est activée pour une tâche, la valeur est `true`. Sinon, la valeur est `false`.

`TaskArn`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

L'exemple suivant affiche les détails renvoyés pour une tâche protégée.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

Les informations suivantes sont renvoyées en cas d'échec.

`Arn`

Amazon Resource Name (ARN) complet de la tâche.

`Detail`

Détails relatifs à l'échec.

`Reason`

Raison de l'échec.

L'exemple suivant affiche les détails renvoyés pour une tâche qui n'est pas protégée.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
        "Detail":null,
        "Reason":"TASK_NOT_VALID"
    }
}
```

Les informations suivantes sont renvoyées en cas d'exception.

`requestID`

Le AWS ID de demande pour l'appel d'API Amazon ECS qui entraîne une exception.

`Arn`

Amazon Resource Name (ARN) complet de la tâche ou du service.

`Code`

Code de l’erreur.

`Message`

Message d’erreur.

> **Note**
>
> Si une erreur `RequestError` ou `RequestTimeout` apparaît, il s'agit probablement d'un problème de mise en réseau. Essayez d'utiliser les points de terminaison d'un VPC pour Amazon ECS.

L'exemple suivant affiche les détails renvoyés en cas d'erreur.

```
{
```

```
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

L'erreur suivante s'affiche si l'agent Amazon ECS ne parvient pas à obtenir de réponse du point de terminaison Amazon ECS pour des raisons telles que des problèmes de réseau ou si le plan de contrôle Amazon ECS est en panne.

```
{
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

L'erreur suivante apparaît lorsque l'agent Amazon ECS reçoit une exception de limitation de la part d'Amazon ECS.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Clusters Amazon ECS pour Fargate

Avec Amazon ECS activé AWS Fargate fournisseurs de capacité, vous pouvez utiliser à la fois la capacité de Fargate et de Fargate Spot pour vos tâches Amazon ECS.

Avec Fargate Spot, vous pouvez exécuter des tâches Amazon ECS tolérantes aux interruptions à un tarif réduit par rapport au prix de Fargate. Fargate Spot exécute les tâches sur la capacité de calcul de réserve. Quand AWS a besoin de retrouver sa capacité, vos tâches sont interrompues avec un avertissement de deux minutes.

Lorsque les tâches qui utilisent les fournisseurs de capacité Fargate et Fargate Spot sont arrêtées, l'événement de changement d'état de la tâche est envoyé à Amazon EventBridge. Le motif de l'arrêt en décrit la cause. Pour plus d'informations, consultez Événements de changement d'état d'une tâche Amazon ECS.

Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité Fargate et de groupe Auto Scaling. Toutefois, une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité Fargate ou de groupe Auto Scaling, mais pas les deux. Pour plus d'informations, consultez la section Fournisseurs de capacité pour les groupes Auto Scaling.

Prenez les points suivants en compte lors de l'utilisation de fournisseurs de capacité :

- Vous devez associer un fournisseur de capacité à un cluster avant de l'associer à la stratégie du fournisseur de capacité.
- Vous pouvez spécifier un maximum de 20 fournisseurs de capacité pour une stratégie de fournisseur de capacité.
- Vous ne pouvez pas mettre à jour un service utilisant un fournisseur de capacité de groupe Auto Scaling pour utiliser un fournisseur de capacité Fargate. L'inverse est également vrai.
- Dans une stratégie de fournisseur de capacité, si aucune valeur `weight` n'est spécifiée pour un fournisseur de capacité dans la console, alors la valeur par défaut `1` est utilisée. Si vous utilisez l'API ou AWS CLI, la valeur par défaut de `0` est utilisé.
- Lorsque plusieurs fournisseurs de capacité sont spécifiés dans le cadre d'une stratégie de fournisseur de capacité, au moins l'un des fournisseurs de capacité doit disposer d'une valeur de poids supérieure à zéro. Les fournisseurs de capacité dont le poids est nul ne sont pas utilisés pour attribuer des tâches. Si vous spécifiez, dans une stratégie, plusieurs fournisseurs de capacité qui possèdent tous un poids nul, toutes les actions `RunTask` ou `CreateService` utilisant la stratégie de fournisseur de capacité échoueront.
- Une valeur de base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité. Si aucune valeur de base n'est spécifiée, la valeur par défaut de zéro est utilisée.
- Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Cependant, une stratégie de fournisseur de

capacité ne peut contenir que des groupes Auto Scaling ou des fournisseurs de capacité Fargate, mais pas les deux.

- Un cluster peut contenir un mélange de services et de tâches autonomes qui utilisent les deux fournisseurs de capacité. Un service peut être mis à jour pour utiliser une stratégie de fournisseur de capacité plutôt qu'un type de lancement. Vous devez toutefois forcer un nouveau déploiement pour ce faire.

Avis de résiliation Spot Fargate

Pendant les périodes de forte demande, la capacité Spot de Fargate peut être indisponible. Cela peut retarder les tâches Spot de Fargate. Lorsque cela se produit, les services Amazon ECS essaient de relancer les tâches jusqu'à ce que la capacité requise soit disponible. Fargate ne remplace pas la capacité Spot par une capacité à la demande.

Lorsque des tâches utilisant la capacité Fargate Spot sont arrêtées en raison d'une interruption Spot, un avertissement d'interruption sous deux minutes est envoyé avant cet arrêt. L'avertissement est envoyé à Amazon en tant qu'événement de changement d'état de la tâche EventBridge et en tant que signal SIGTERM à la tâche en cours. Si vous utilisez Fargate Spot au titre d'un service, le planificateur de service reçoit dans ce scénario le signal d'interruption et tente de lancer des tâches supplémentaires sur Fargate Spot si la capacité est disponible. Un service avec une seule tâche est interrompu jusqu'à ce que la capacité soit disponible. Pour plus d'informations sur un arrêt progressif, veuillez consulter le billet de blog Graceful shutdowns with ECS.

Pour faire en sorte que vos conteneurs se ferment correctement avant l'arrêt de la tâche, vous pouvez configurer les éléments suivants :

- Il est possible de spécifier une valeur de temporisation d'arrêt (`stopTimeout`) de `120` secondes ou moins dans la définition de conteneur utilisée par la tâche. La valeur `stopTimeout` par défaut est de 30 secondes. Vous pouvez spécifier une valeur `stopTimeout` plus longue pour disposer de plus de temps entre le moment où l'événement de changement d'état de tâche est reçu et l'instant où le conteneur est arrêté de force. Pour plus d'informations, consultez Temporisations de conteneurs.
- Le signal SIGTERM doit être reçu à l'intérieur du conteneur pour effectuer des actions de nettoyage. Si vous ne parvenez pas à traiter ce signal, la tâche reçoit un signal SIGKILL après le `stopTimeout` configuré et peut entraîner une perte ou une corruption de données.

Voici un extrait d'événement de changement d'état de tâche. Cet extrait indique le motif et le code d'arrêt d'une interruption Fargate Spot.

```
{
  "version": "0",
  "id": "9bcdac79-b31f-4d3d-9410-fbd727c29fab",
  "detail-type": "ECS Task State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:task/b99d40b3-5176-4f71-9a52-9dbd6f1cebef"
  ],
  "detail": {
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "createdAt": "2016-12-06T16:41:05.702Z",
    "desiredStatus": "STOPPED",
    "lastStatus": "RUNNING",
    "stoppedReason": "Your Spot Task was interrupted.",
    "stopCode": "SpotInterruption",
    "taskArn": "arn:aws:ecs:us-east-1:111122223333:task/
b99d40b3-5176-4f71-9a52-9dbd6fEXAMPLE",
    ...
  }
}
```

Voici un modèle d'événement utilisé pour créer un EventBridge règle pour les événements de changement d'état des tâches Amazon ECS. Vous pouvez éventuellement spécifier un cluster dans le champ `detail`. Cela signifie que vous recevrez des événements de changement d'état de tâche pour ce cluster. Pour plus d'informations sur la création d'un EventBridge règle, voir Premiers pas avec Amazon EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Task State Change"
    ],
    "detail": {
        "clusterArn": [
            "arn:aws:ecs:us-west-2:111122223333:cluster/default"
        ]
```

```
    }
}
```

Création d'un cluster Amazon ECS pour les charges de travail Fargate

Vous créez un cluster pour définir l'infrastructure sur laquelle vos tâches et vos services s'exécutent.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour plus d'informations, consultez the section called "Exemples de cluster Amazon ECS". La console Amazon ECS crée les ressources nécessaires à un cluster Amazon ECS en créant un CloudFormation empiler.

La console associe automatiquement les fournisseurs de capacité Fargate et Fargate Spot au cluster.

Vous pouvez modifier les options suivantes :

- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour plus d'informations, consultez Interconnexion des services Amazon ECS.
- Activer la réception d'événements liés aux tâches EventBridge notifications pour les changements d'état des tâches.
- Ajoutez des balises pour vous aider à identifier vos clusters.
- Attribuer un AWS KMS clé pour votre stockage géré. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Attribuer un AWS KMS clé pour votre stockage éphémère Fargate. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Configurez le AWS KMS clé et journalisation pour ECS Exec.

Procédure

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.

3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :

   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, sous Paramètres par défaut de Service Connect, pour Espace de noms par défaut, sélectionnez ou saisissez un nom d'espace de noms. Pour utiliser un espace de noms partagé, choisissez ou saisissez un ARN d'espace de noms. Pour plus d'informations sur l'utilisation d'espaces de nom partagés, consultez la section Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms.
6. (Facultatif) Utilisez Container Insights, développez Surveillance, puis sélectionnez une des options suivantes :

   - Pour utiliser Container Insights avec observabilité améliorée comme recommandé, sélectionnez Container Insights avec observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
7. (Facultatif) Pour activer les événements liés aux tâches, développez Événements liés aux tâches, puis allumez Activer les événements liés aux tâches.

   Lorsque vous activez les événements de tâches, Amazon ECS envoie les événements de changement d'état des tâches à EventBridge. Cela vous permet de surveiller les modifications du cycle de vie des tâches et d'y répondre automatiquement.
8. (Facultatif) Pour utiliser ECS Exec afin de déboguer les tâches dans le cluster, développez Configuration de la résolution des problèmes, puis configurez les éléments suivants :

   - (Facultatif) Pour AWS KMS clé pour ECS Exec, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de session ECS Exec.
   - (Facultatif) Pour Journalisation ECS Exec, choisissez la destination du journal :
     - Pour envoyer des journaux à CloudWatch Journaux, choisissez Amazon CloudWatch.
     - Pour envoyer les journaux à Amazon S3, choisissez Amazon S3.
     - Pour désactiver la journalisation, choisissez Aucune.
9. (Facultatif). Sous Chiffrement, vous pouvez configurer les éléments suivants :

- Chiffrer vos données sur le stockage éphémère Fargate. En dessous Chiffrement, pour Stockage éphémère Fargate, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour crypter les données de stockage éphémères de Fargate.
- Chiffrer les données sur le stockage géré. En dessous Chiffrement, pour Stockage géré, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de stockage gérées.

10. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

    [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

    - Pour Key (Clé), saisissez le nom de la clé.
    - Pour Value (Valeur), saisissez la valeur de clé.

    [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
11. Choisissez Créer.

Étapes suivantes

Après avoir créé le cluster, vous pouvez créer des définitions de tâches pour vos applications, puis les exécuter en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d’informations, consultez les ressources suivantes :

- Définitions de tâche Amazon ECS
- Exécution d’une application en tant que tâche Amazon ECS
- Création d’un déploiement de mise à jour propagée Amazon ECS

Fournisseurs de capacité Amazon ECS pour les charges de travail EC2

Lorsque vous utilisez les instances Amazon EC2 pour votre capacité, vous utilisez les groupes Auto Scaling pour gérer les instances Amazon EC2 enregistrées sur leurs clusters. L’autoscaling permet de vous assurer que vous disposez du bon nombre d’instances Amazon EC2 disponibles pour gérer la charge de l’application.

Vous pouvez utiliser la fonctionnalité de mise à l'échelle gérée pour qu'Amazon ECS gère l'augmentation et la réduction horizontale du groupe Auto Scaling, ou vous pouvez gérer les actions de mise à l'échelle vous-même. Pour plus d'informations, consultez Gestion automatique de la capacité Amazon ECS grâce à l'autoscaling de cluster.

Nous vous recommandons de créer un groupe Auto Scaling vide. Si vous utilisez un groupe Auto Scaling existant, il se peut que les instances Amazon EC2 associées au groupe qui étaient déjà en cours d'exécution et enregistrées sur un cluster Amazon ECS avant le groupe Auto Scaling utilisé pour créer un fournisseur de capacité ne soient pas bien enregistrées auprès du fournisseur de capacité. Cela peut occasionner des problèmes lorsque le fournisseur de capacité est utilisé dans une stratégie de fournisseur de capacité. Utilisez `DescribeContainerInstances` pour vérifier qu'une instance de conteneur est bien associée à un fournisseur de capacité.

> **Note**
>
> Pour créer un groupe Auto Scaling vide, réglez le nombre souhaité sur zéro. Après avoir créé le fournisseur de capacité et l'avoir associé à un cluster, vous pouvez le mettre à l'échelle. Lorsque vous utilisez la console Amazon ECS, Amazon ECS crée un modèle de lancement Amazon EC2 et un groupe Auto Scaling en votre nom dans le cadre du CloudFormation empiler. Ils sont préfixés par `EC2ContainerService-<ClusterName>`. Vous pouvez utiliser le groupe Auto Scaling comme fournisseur de capacité pour ce cluster.

Nous vous recommandons d'utiliser le drainage d'instance gérée pour permettre la résiliation progressive des instances Amazon EC2 sans perturber vos charges de travail. Cette fonctionnalité est activée par défaut. Pour plus d'informations, consultez Arrêt en toute sécurité des charges de travail Amazon ECS exécutées sur les instances EC2

Les points suivants doivent être pris en compte lors de l'utilisation de fournisseurs de capacité de groupe Auto Scaling dans la console :

- La valeur du paramètre `MaxSize` d'un groupe Auto Scaling doit être supérieure à zéro pour une montée en puissance.
- Le groupe Auto Scaling ne peut pas avoir de paramètres de pondération d'instance.
- Si le groupe Auto Scaling ne peut pas monter en puissance pour s'adapter au nombre de tâches exécutées, les tâches ne parviennent pas à passer outre l'état `PROVISIONING`.
- Ne modifiez pas la ressource de politique de mise à l'échelle associée à vos groupes Auto Scaling gérés par des fournisseurs de capacité.

- Si la mise à l'échelle gérée est activée au moment où vous créez un fournisseur de capacité, le nombre souhaité de groupes Auto Scaling peut être défini sur `0`. Lorsque la mise à l'échelle gérée est activée, Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance du groupe Auto Scaling.
- Vous devez associer un fournisseur de capacité à un cluster avant de l'associer à la stratégie du fournisseur de capacité.
- Vous pouvez spécifier un maximum de 20 fournisseurs de capacité pour une stratégie de fournisseur de capacité.
- Vous ne pouvez pas mettre à jour un service utilisant un fournisseur de capacité de groupe Auto Scaling pour utiliser un fournisseur de capacité Fargate. L'inverse est également vrai.
- Dans une stratégie de fournisseur de capacité, si aucune valeur `weight` n'est spécifiée pour un fournisseur de capacité dans la console, alors la valeur par défaut `1` est utilisée. Si vous utilisez l'API ou AWS CLI, la valeur par défaut de `0` est utilisé.
- Lorsque plusieurs fournisseurs de capacité sont spécifiés dans le cadre d'une stratégie de fournisseur de capacité, au moins l'un des fournisseurs de capacité doit disposer d'une valeur de poids supérieure à zéro. Les fournisseurs de capacité dont le poids est nul ne sont pas utilisés pour attribuer des tâches. Si vous spécifiez, dans une stratégie, plusieurs fournisseurs de capacité qui possèdent tous un poids nul, toutes les actions `RunTask` ou `CreateService` utilisant la stratégie de fournisseur de capacité échoueront.
- Une valeur de base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité. Si aucune valeur de base n'est spécifiée, la valeur par défaut de zéro est utilisée.
- Un cluster peut contenir une combinaison à la fois de fournisseurs de capacité de groupe Auto Scaling et de fournisseurs de capacité Fargate. Toutefois, une stratégie de fournisseur de capacité ne peut contenir que des fournisseurs de capacité du groupe Auto Scaling ou Fargate, mais pas les deux.
- Un cluster peut contenir une combinaison de services et de tâches autonomes utilisant à la fois des fournisseurs de capacité et des types de lancement. Un service peut être mis à jour pour utiliser une stratégie de fournisseur de capacité plutôt qu'un type de lancement. Vous devez toutefois forcer un nouveau déploiement pour ce faire.
- Amazon ECS prend en charge les groupes d'instances pré-initialisées Amazon EC2 Auto Scaling. Un groupe d'instances pré-initialisées est un groupe d'instances Amazon EC2 pré-initialisées prêtes à être mises en service. Chaque fois que votre application doit être mise à l'échelle, Amazon EC2 Auto Scaling utilise les instances préinitialisées du groupe d'instances préinitialisées plutôt que de lancer des instances non initialisées. Cela permet à tout processus d'initialisation final

de s’exécuter avant que l’instance ne soit mise en service. Pour plus d’informations, consultez Configuration d’instances préinitialisées pour votre groupe Auto Scaling Amazon ECS.

Pour plus d’informations sur la création de modèles de lancement pour Amazon EC2 Auto Scaling, consultez la section Modèles de lancement dans le Guide de l’utilisateur Amazon EC2 Auto Scaling. Pour plus d'informations sur la création de groupe Amazon EC2 Auto Scaling, consultez la section Création de groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Considérations relatives à la sécurité des instances de conteneur Amazon EC2 pour Amazon ECS

Vous devez envisager une instance de conteneur unique et son accès dans le cadre de votre modèle de menace. Par exemple, une seule tâche affectée peut être en mesure de tirer parti des autorisations IAM d'une tâche non infectée sur la même instance.

Nous vous recommandons d'utiliser la procédure suivante pour empêcher cela :

- N'utilisez pas les privilèges d'administrateur lors de l'exécution de vos tâches.
- Attribuez un rôle de tâche avec un accès de moindre privilège à vos tâches.

  L'agent de conteneur crée automatiquement un jeton avec un ID d'informations d'identification unique qui est utilisé pour accéder aux ressources Amazon ECS.
- Pour empêcher les conteneurs exécutés par des tâches utilisant le mode réseau `awsvpc` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, définissez la variable de configuration d'agent `ECS_AWSVPC_BLOCK_IMDS` sur true dans le fichier de configuration de l'agent et redémarrez l'agent.
- Utilisez Amazon GuardDuty Surveillance de l'exécution pour détecter les menaces visant les clusters et les conteneurs au sein de votre AWS environnement. Runtime Monitoring utilise un GuardDuty agent de sécurité qui ajoute de la visibilité à l'exécution des charges de travail Amazon ECS individuelles, par exemple, l'accès aux fichiers, l'exécution des processus et les connexions réseau. Pour plus d'informations, voir GuardDuty Surveillance du temps d'exécution dans le GuardDuty Guide de l'utilisateur.

Création d’un cluster Amazon ECS pour les charges de travail Amazon EC2

Vous créez un cluster pour définir l’infrastructure sur laquelle vos tâches et vos services s’exécutent.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l'autorisation IAM appropriée. Pour plus d'informations, consultez the section called "Exemples de cluster Amazon ECS". La console Amazon ECS fournit un moyen simple de créer les ressources nécessaires à un cluster Amazon ECS en créant un CloudFormation empiler.

Pour simplifier au maximum le processus de création de clusters, la console propose des sélections par défaut pour de nombreux choix, que nous décrivons dans les sections suivantes. La plupart des sections de la console comportent des volets d'aide qui fournissent un contexte supplémentaire.

Vous pouvez enregistrer des instances Amazon EC2 lors de la création du cluster ou enregistrer des instances supplémentaires avec le cluster après sa création.

Vous pouvez modifier les options par défaut suivantes :

- Modifier les sous-réseaux sur lesquels vos instances sont lancées.
- Modifier les groupes de sécurité utilisés pour contrôler le trafic vers les instances de conteneur.
- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour plus d'informations, consultez Interconnexion des services Amazon ECS.
- Activer la réception d'événements liés aux tâches EventBridge notifications pour les changements d'état des tâches.
- Attribuer un AWS KMS clé pour votre stockage géré. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Attribuer un AWS KMS clé pour votre stockage éphémère Fargate. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Configurez le AWS KMS clé et journalisation pour ECS Exec.
- Ajoutez des balises pour vous aider à identifier vos clusters.

Options de groupe Auto Scaling

Lorsque vous utilisez des instances Amazon EC2, vous devez spécifier un groupe Auto Scaling pour gérer l'infrastructure sur laquelle vos tâches et services s'exécutent.

Lorsque vous choisissez de créer un groupe Auto Scaling, il est automatiquement configuré pour le comportement suivant :

- Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance du groupe Auto Scaling.
- Amazon ECS empêche la résiliation des instances Amazon EC2 d'un groupe Auto Scaling contenant des tâches lors d'une action de mise à l'échelle horizontale. Pour plus d'informations, consultez Protection des instances dans le Guide de l'utilisateur AWS Auto Scaling .

Vous configurez les propriétés de groupe Auto Scaling suivantes qui déterminent le type et le nombre d'instances à lancer pour le groupe :

- L'Amazonie ECS-optimized AMI.
- Type d'instance.
- La paire de clés SSH qui prouve votre identité lorsque vous connectez à l'instance. Pour de plus amples informations sur comment créer des clés SSH, consultez la section Paires de clés Amazon EC2 et instances Linux dans le Guide de l'utilisateur Amazon EC2.
- Nombre minimum d'instances à lancer pour le groupe Auto Scaling.
- Nombre maximal d'instances démarrées pour le groupe Auto Scaling.

  Pour que le groupe puisse monter en puissance, le maximum doit être supérieur à 0.

Amazon ECS crée un modèle de lancement Amazon EC2 Auto Scaling et un groupe Auto Scaling en votre nom dans le cadre de la pile CloudFormation . Les valeurs que vous avez spécifiées pour l'AMI, les types d'instances et la paire de clés SSH se trouvent dans le modèle de lancement. Les modèles sont précédés du préfixe `EC2ContainerService-<ClusterName>`, ce qui facilite leur identification. Les groupes Auto Scaling sont précédés du préfixe `<ClusterName>-ECS-Infra-ECSAutoScalingGroup`.

Les instances lancées pour le groupe Auto Scaling utilisent le modèle de lancement.

Options de mise en réseau

Par défaut, les instances sont lancées dans les sous-réseaux par défaut de la région. Les groupes de sécurité qui contrôlent le trafic vers vos instances de conteneur, actuellement associés aux sous-réseaux, sont utilisés. Vous pouvez modifier les sous-réseaux et les groupes de sécurité des instances.

Vous pouvez choisir un sous-réseau existant. Vous pouvez soit utiliser un groupe de sécurité existant, soit en créer un. Pour créer des tâches dans un IPv6-only configuration, utilisez des sous-réseaux qui incluent uniquement un bloc d'adresse CIDR IPv6.

Lorsque vous créez un groupe de sécurité, vous devez spécifier au moins une règle entrante.

Les règles entrantes déterminent le trafic qui peut atteindre vos instances de conteneur et incluent les propriétés suivantes :

- Le protocole à autoriser
- La gamme de ports à autoriser
- Le trafic entrant (source)

Pour autoriser le trafic entrant provenant d'une adresse ou d'un bloc d'adresses CIDR spécifique, utilisez Personnalisée pour Source avec le CIDR autorisé.

Pour autoriser le trafic entrant en provenance de toutes les destinations, utilisez Partout pour Source. Cela ajoute automatiquement la valeur 0.0.0.0/0 Bloc CIDR IPv4 et bloc CIDR IPv6 : :/0.

Pour autoriser le trafic entrant depuis votre ordinateur local, utilisez Groupe source pour Source. Cette action ajoute automatiquement l'adresse IP actuelle de votre ordinateur local comme source autorisée.

Pour créer un nouveau cluster (console Amazon ECS)

Avant de commencer, attribuez l'autorisation IAM adéquate. Pour plus d’informations, consultez the section called “Exemples de cluster Amazon ECS”.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :

    - Pour Nom du cluster, saisissez un nom unique.

      Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.

- (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, sous Paramètres par défaut de Service Connect, pour Espace de noms par défaut, sélectionnez ou saisissez un nom d'espace de noms. Pour utiliser un espace de noms partagé, choisissez ou saisissez un ARN d'espace de noms. Pour plus d'informations sur l'utilisation d'espaces de nom partagés, consultez la section Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms.

6. Ajoutez des instances Amazon EC2 à votre cluster, développez Infrastructures puis sélectionnez Fargate et Self-managed les instances.

   Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité :

   a. Pour utiliser un groupe Auto Scaling existant, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez le groupe.

   b. Pour créer un groupe Auto Scaling, depuis Auto Scaling group (ASG) (Groupe Auto Scaling [ASG]), sélectionnez Create new group (Créer un nouveau groupe), puis fournissez les détails suivants sur le groupe :

      - Pour Modèle de provisionnement, choisissez si vous souhaitez utiliser On-demand instances ou Spot Instances.
      - Si vous choisissez d'utiliser des instances Spot, pour Stratégie d'allocation, sélectionnez les groupes de capacité Spot (types d'instances et zones de disponibilité) à utiliser pour les instances.

        Pour la plupart des charges de travail, vous pouvez choisir Capacité et prix optimisés.

        Pour plus d'informations, voir Stratégies d'allocation pour les instances Spot dans le Guide de l'utilisateur Amazon EC2.
      - Pour Instance de conteneur Amazon Machine Image (AMI), choisissez Amazon ECS-optimized AMI pour les instances du groupe Auto Scaling.
      - Pour EC2 instance type (Type d'instance EC2), choisissez le type d'instance pour vos charges de travail.

        La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.
      - Pour Rôle d'instance EC2, choisissez un rôle d'instance de conteneur existant ou créez-en un autre.

Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

- Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling.
- Pour SSH key pair (Paire de clés SSH), choisissez la paire qui prouve votre identité lorsque vous connectez à l'instance.
- Pour permettre une image et un stockage plus grands, pour Taille du volume EBS racine, saisissez la valeur en Gio.

7. (Facultatif) Pour modifier le VPC et les sous-réseaux, sous Mise en réseau pour les instances Amazon EC2, effectuez l'une des opérations suivantes :

- Pour supprimer un sous-réseau, sous Subnets (Sous-réseaux), choisissez X pour chaque sous-réseau que vous souhaitez supprimer.
- Pour passer à un VPC autre que celui par défaut, sous VPC, choisissez un VPC existant, puis sous Sous-réseaux, sélectionnez les sous-réseaux. Pour un IPv6-only configuration, choisissez un VPC doté d'un bloc d'adresse CIDR IPv6 et des sous-réseaux dotés uniquement d'un bloc d'adresse CIDR IPv6.
- Choisissez les groupes de sécurité. Sous Groupe de sécurité, choisissez l'une des options suivantes :
  - Pour utiliser un groupe de sécurité existant, choisissez Utiliser un groupe de sécurité existant, puis sélectionnez le groupe de sécurité.
  - Pour créer un groupe de sécurité, sélectionnez Créer un nouveau groupe de sécurité. Ensuite, choisissez Ajouter une règle pour chaque règle entrante.

    Pour plus d'informations sur les règles entrantes, veuillez consulter Options de mise en réseau.
- Pour attribuer automatiquement des adresses IP publiques à vos instances de conteneur Amazon EC2, pour Auto-assign IP publique, choisissez l'une des options suivantes :
  - Utiliser le paramètre de sous-réseau : attribuez une adresse IP publique aux instances lorsque le sous-réseau dans lequel les instances sont lancées est un sous-réseau public.
  - Activer : attribuez une adresse IP publique aux instances.

8. (Facultatif) Utilisez Container Insights, développez Surveillance, puis sélectionnez une des options suivantes :

- Pour utiliser Container Insights avec observabilité améliorée comme recommandé, sélectionnez Container Insights avec observabilité améliorée.
- Pour utiliser Container Insights, choisissez Container Insights.

9. (Facultatif) Pour activer les événements liés aux tâches, développez Événements liés aux tâches, puis allumez Activer les événements liés aux tâches.

   Lorsque vous activez les événements de tâches, Amazon ECS envoie les événements de changement d'état des tâches à EventBridge. Cela vous permet de surveiller les modifications du cycle de vie des tâches et d'y répondre automatiquement.

10. (Facultatif) Pour utiliser ECS Exec afin de déboguer les tâches dans le cluster, développez Configuration de la résolution des problèmes, puis configurez les éléments suivants :

    - (Facultatif) Pour AWS KMS clé pour ECS Exec, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de session ECS Exec.
    - (Facultatif) Pour Journalisation ECS Exec, choisissez la destination du journal :
      - Pour envoyer des journaux à CloudWatch Journaux, choisissez Amazon CloudWatch.
      - Pour envoyer les journaux à Amazon S3, choisissez Amazon S3.
      - Pour désactiver la journalisation, choisissez Aucune.

11. (Facultatif)

    Si vous utilisez Runtime Monitoring avec l'option manuelle et que vous souhaitez que ce cluster soit surveillé par GuardDuty, choisissez Ajouter un tag et procédez comme suit :

    - Pour Clé, saisissez **`guardDutyRuntimeMonitoringManaged`**.
    - Pour le champ Valeur, saisissez **`true`**.

12. (Facultatif) Chiffrez les données sur le stockage géré. En dessous Chiffrement, pour Stockage géré, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de stockage gérées.

13. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

    [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

    - Pour Key (Clé), saisissez le nom de la clé.
    - Pour Value (Valeur), saisissez la valeur de clé.

[Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.

14. Choisissez Créer.

Étapes suivantes

Après avoir créé le cluster, vous pouvez créer des définitions de tâches pour vos applications, puis les exécuter en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d'informations, consultez les ressources suivantes :

- Définitions de tâche Amazon ECS
- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mise à jour propagée Amazon ECS

Gestion automatique de la capacité Amazon ECS grâce à l'autoscaling de cluster

Amazon ECS peut gérer la mise à l'échelle des instances Amazon EC2 enregistrées sur votre cluster. C'est ce qu'on appelle l'autoscaling du cluster Amazon ECS. Vous activez la mise à l'échelle gérée lors de la création du fournisseur de capacité de groupe Auto Scaling Amazon ECS. Vous définissez ensuite un pourcentage cible (la `targetCapacity`) pour l'utilisation de l'instance dans ce groupe Auto Scaling. Amazon ECS crée deux versions personnalisées CloudWatch des mesures et une politique de mise à l'échelle du suivi des cibles pour votre groupe Auto Scaling. Amazon ECS gère ensuite l'évolutivité horizontale en fonction des ressources que vos tâches utilisent.

Pour chaque fournisseur de capacité de groupe Auto Scaling associé à un cluster, Amazon ECS crée et gère les ressources suivantes :

- Une valeur métrique faible CloudWatch alarme
- Une valeur métrique élevée CloudWatch alarme
- Stratégie de mise à l'échelle de suivi cible

> **Note**
>
> Amazon ECS crée la politique de mise à l'échelle de suivi cible et la joint au groupe Auto Scaling. Pour mettre à jour la stratégie de dimensionnement de suivi cible, mettez à jour les

paramètres de dimensionnement géré du fournisseur de capacité au lieu de mettre à jour directement la politique de dimensionnement.

Lorsque vous désactivez le dimensionnement géré ou que vous dissociez le fournisseur de capacité d'un cluster, Amazon ECS supprime les deux CloudWatch les mesures et le suivi de la cible, les ressources de la politique de dimensionnement.

Amazon ECS utilise les métriques suivantes pour déterminer les mesures à prendre :

`CapacityProviderReservation`

Le pourcentage d'instances de conteneur utilisées pour un fournisseur de capacité spécifique. Amazon ECS génère cette métrique.

Amazon ECS définit la valeur `CapacityProviderReservation` sur un nombre compris entre 0 et 100. Amazon ECS utilise la formule suivante pour représenter le ratio de capacité restant dans le groupe Auto Scaling. Amazon ECS publie ensuite la métrique pour CloudWatch. Pour plus d'informations sur le calcul de la métrique, consultez la section le billet de blog Deep Dive on Amazon ECS Cluster Auto Scaling.

```
CapacityProviderReservation = (number of instances needed) / (number of running
 instances) x 100
```

`DesiredCapacity`

Quantité de capacité pour le groupe Auto Scaling. Cette métrique n'est pas publiée sur CloudWatch.

Amazon ECS publie le `CapacityProviderReservation` métrique à CloudWatch dans le `AWS/ECS/ManagedScaling` espace de noms. La métrique `CapacityProviderReservation` entraîne l'une des actions suivantes :

La valeur **`CapacityProviderReservation`** est égale à **`targetCapacity`**.

Le groupe Auto Scaling n'a pas besoin d'une mise à l'échelle horizontale ou d'une montée en puissance. Le pourcentage d'utilisation cible a été atteint.

La valeur **`CapacityProviderReservation`** est supérieure à **`targetCapacity`**.

Un plus grand nombre de tâches utilisent un pourcentage de capacité supérieur à votre pourcentage `targetCapacity`. L'augmentation de la valeur de `CapacityProviderReservation` la métrique provoque les CloudWatch alarme pour agir. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Le groupe Auto Scaling utilise cette valeur pour lancer des instances EC2, puis les enregistrer auprès du cluster.

Lorsque la valeur par défaut de `targetCapacity` est de 100 %, les nouvelles tâches sont à l'état `PENDING` pendant la montée en puissance, car les instances ne disposent pas de la capacité nécessaire pour exécuter les tâches. Une fois les nouvelles instances enregistrées auprès d'ECS, ces tâches commenceront sur les nouvelles instances.

La valeur **`CapacityProviderReservation`** est inférieure à **`targetCapacity`**.

Il y a moins de tâches utilisant un pourcentage de capacité inférieur à votre `targetCapacity` pourcentage et il y a au moins une instance qui peut être supprimée. La valeur réduite de `CapacityProviderReservation` la métrique provoque les CloudWatch alarme pour agir. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Le groupe Auto Scaling utilise cette valeur pour résilier des instances de conteneur EC2, puis annuler leur enregistrement dans le cluster.

Le groupe Auto Scaling suit les stratégies de résiliation du groupe pour déterminer quelles instances seront résiliées en premier lors d'événements de mise à l'échelle horizontale. De plus, cela évite les instances ayant un paramètre de protection contre la mise à l'échelle horizontale d'instance activé. L'autoscaling de cluster permet de gérer les instances dotées du paramètre de protection contre la mise à l'échelle horizontale des instances si vous activez la protection contre la résiliation gérée. Pour plus d'informations sur la protection contre la résiliation gérée, veuillez consulter Contrôle des instances résiliées par Amazon ECS. Pour plus d'informations sur la résiliation des instances par les groupes Auto Scaling, veuillez consulter Contrôle des instances Auto Scaling à résilier lors d'une mise à l'échelle horizontale dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Les points suivants doivent être pris en compte lors de l'utilisation de l'autoscaling de cluster :

- Ne modifiez pas ou ne gérez pas la capacité souhaitée pour le groupe Auto Scaling associé à un fournisseur de capacité avec d'autres stratégies de mise à l'échelle que celle gérée par Amazon ECS.

- Lorsqu'Amazon ECS passe à zéro instance, il lance automatiquement deux instances.
- Amazon ECS utilise le `AWSServiceRoleForECS` rôle IAM lié à un service pour les autorisations dont il a besoin pour appeler AWS Auto Scaling en votre nom. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- Lorsque vous utilisez des fournisseurs de capacité avec des groupes Auto Scaling, l'utilisateur, le groupe ou le rôle qui crée les fournisseurs de capacité a besoin de l'autorisation `autoscaling:CreateOrUpdateTags`. En effet, Amazon ECS ajoute une balise au groupe Auto Scaling lorsqu'il l'associe au fournisseur de capacité.

  > **Important**
  >
  > Assurez-vous que les outils que vous utilisez ne suppriment pas la balise `AmazonECSManaged` du groupe Auto Scaling. Si cette balise est supprimée, Amazon ECS ne peut pas gérer la mise à l'échelle.

- La mise à l'échelle automatique du cluster ne modifie pas le MinimumCapacity ou MaximumCapacity pour le groupe. Pour que le groupe s'agrandisse, la valeur de MaximumCapacity doit être supérieur à zéro.
- Lorsque la fonction Auto Scaling (mise à l'échelle gérée) est activée, un fournisseur de capacité ne peut être connecté qu'à un seul cluster à la fois. Si votre fournisseur de capacité a désactivé la mise à l'échelle gérée, vous pouvez l'associer à plusieurs clusters.
- Lorsque la mise à l'échelle gérée est désactivée, le fournisseur de capacité ne bénéficie pas d'une mise à l'échelle horizontale ou d'une montée en puissance. Vous pouvez utiliser une stratégie de fournisseur de capacité pour équilibrer vos tâches entre les fournisseurs de capacité.
- La stratégie `binpack` est la plus efficace en termes de capacité.
- Lorsque la capacité cible est inférieure à 100 %, dans le cadre de la stratégie de placement, la stratégie `binpack` doit avoir un ordre supérieur à la stratégie `spread`. Cela empêche l'augmentation horizontale du fournisseur de capacité jusqu'à ce que chaque tâche dispose d'une instance dédiée ou que la limite soit atteinte.

Activation de l'autoscaling de cluster

Vous pouvez activer l'autoscaling de cluster à l'aide de la console ou de l' AWS CLI.

Lorsque vous créez un cluster qui utilise des fournisseurs de capacité EC2 à l'aide de la console, Amazon ECS crée un groupe Auto Scaling en votre nom et définit la capacité cible. Pour

plus d'informations, consultez Création d'un cluster Amazon ECS pour les charges de travail Amazon EC2.

Vous pouvez également créer un groupe Auto Scaling, puis l'attribuer à un cluster. Pour plus d'informations, consultez Mise à jour d'un fournisseur de capacité Amazon ECS.

Lorsque vous utilisez AWS CLI, après avoir créé le cluster

1. Avant de créer le fournisseur de capacité, vous devez créer un groupe Auto Scaling. Pour plus d'informations, voir Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.
2. Utilisez `put-cluster-capacity-providers` pour modifier le fournisseur de capacité du cluster. Pour plus d'informations, consultez Activation de l'autoscaling de cluster Amazon ECS.

Optimisation de l'autoscaling de cluster Amazon ECS

Les clients qui exécutent Amazon ECS sur Amazon EC2 peuvent tirer parti de l'autoscaling de cluster pour gérer la mise à l'échelle des groupes Amazon EC2 Auto Scaling. Grâce à l'autoscaling de cluster, vous pouvez configurer Amazon ECS pour qu'il mette automatiquement à l'échelle votre groupe Auto Scaling, et vous concentrer uniquement sur l'exécution de vos tâches. Amazon ECS veille à ce que le groupe Auto Scaling évolue horizontalement en fonction des besoins, sans qu'aucune autre intervention ne soit requise. Les fournisseurs de capacité Amazon ECS vous permettent de gérer l'infrastructure de votre cluster en s'assurant qu'il y a suffisamment d'instances de conteneur pour répondre aux demandes de votre application. Pour savoir comment fonctionne l'autoscaling de cluster, consultez le billet de blog Deep Dive on Amazon ECS Cluster Auto Scaling.

La mise à l'échelle automatique des clusters repose sur une CloudWatch intégration basée sur le groupe Auto Scaling pour ajuster la capacité du cluster. Par conséquent, il a une latence inhérente associée :

- Publier le CloudWatch des indicateurs,
- Le temps nécessaire pour la métrique `CapacityProviderReservation` violer CloudWatch alarmes (hautes et basses)
- au temps nécessaire à la préinitialisation d'une instance Amazon EC2 récemment lancée. Vous pouvez prendre les mesures suivantes pour améliorer la réactivité de l'autoscaling de cluster et accélérer les déploiements :

Mise à l'échelle par étapes du fournisseur de capacités

Les fournisseurs de capacité Amazon ECS devront grow/shrink les instances de conteneur pour répondre aux exigences de votre application. Le nombre minimum d'instances qu'Amazon ECS lancera est fixé à une par défaut. Cela peut allonger la durée de vos déploiements si plusieurs instances sont nécessaires pour placer vos tâches en attente. Vous pouvez augmenter la `minimumScalingStepSize` via l'API Amazon ECS afin d'accroître le nombre minimum d'instances qu'Amazon ECS peut faire évoluer horizontalement à la fois. Une `maximumScalingStepSize` trop faible peut limiter le nombre d'instances de conteneur pouvant être mises à l'échelle horizontale, ce qui peut ralentir vos déploiements.

> **Note**
>
> Cette configuration n'est actuellement disponible que via les API `CreateCapacityProvider` ou `UpdateCapacityProvider`.

Durée de préinitialisation de l'instance

La période de préchauffage de l'instance est la période au terme de laquelle une instance Amazon EC2 récemment lancée peut contribuer à CloudWatch métriques pour le groupe Auto Scaling. Une fois la période de préinitialisation spécifiée expirée, l'instance est prise en compte dans les mesures agrégées du groupe Auto Scaling, et l'autoscaling de cluster poursuit sa prochaine itération de calculs pour estimer le nombre d'instances requises.

La valeur par défaut pour `instanceWarmupPeriod` est de 300 secondes, que vous pouvez configurer à une valeur inférieure via les API `CreateCapacityProvider` ou `UpdateCapacityProvider` pour une mise à l'échelle plus réactive. Nous vous recommandons de définir la valeur sur une valeur supérieure à 60 secondes afin d'éviter la surallocation.

Capacité de réserve

Si votre fournisseur de capacité ne dispose d'aucune instance de conteneur pour placer des tâches, il doit accroître (augmenter horizontalement) la capacité du cluster en lançant des instances Amazon EC2 à la volée, puis attendre leur démarrage avant de pouvoir y lancer des conteneurs. Cela peut réduire considérablement la vitesse de lancement des tâches. Vous avez deux options.

Dans ce cas, le fait de disposer d'une capacité Amazon EC2 disponible déjà lancée et prête à exécuter des tâches augmentera la vitesse de lancement effective des tâches. Vous pouvez utiliser la configuration de `Target Capacity` pour indiquer que vous souhaitez conserver de la capacité de

réserve dans vos clusters. Par exemple, en définissant `Target Capacity` sur 80 %, vous indiquez que votre cluster a besoin de 20 % de capacité de réserve à tout moment. Cette capacité de réserve peut permettre le lancement immédiat de toutes les tâches autonomes, garantissant ainsi que les lancements de tâches ne sont pas ralentis. L'inconvénient de cette approche est l'augmentation potentielle des coûts liés au maintien de la capacité de réserve des clusters.

Une autre approche que vous pouvez envisager consiste à augmenter la marge de manœuvre de votre service, et non celle du fournisseur de capacité. Cela signifie qu'au lieu de réduire la configuration de `Target Capacity` pour lancer de la capacité de réserve, vous pouvez augmenter le nombre de réplicas dans votre service en modifiant la métrique de mise à l'échelle du suivi de cible ou les seuils de mise à l'échelle par étapes de l'autoscaling du service. Notez que cette approche ne sera utile que pour les charges de travail complexes, mais n'aura aucun effet lorsque vous déployez de nouveaux services et que vous passez de 0 à N tâches pour la première fois. Pour plus d'informations sur les politiques de mise à l'échelle associées, consultez les sections Politiques de mise à l'échelle du suivi de cible ou Politiques de mise à l'échelle par étapes dans le Guide du développeur Amazon Elastic Container Service.

Comportement de mise à l'échelle gérée par Amazon ECS

Lorsque vous disposez de fournisseurs de capacité de groupe Auto Scaling qui utilisent la mise à l'échelle gérée, Amazon ECS estime le nombre optimal d'instances à ajouter à votre cluster et utilise cette valeur pour déterminer le nombre d'instances à demander ou à libérer.

Comportement de montée en puissance gérée

Amazon ECS sélectionne un fournisseur de capacité pour chaque tâche en appliquant la stratégie du fournisseur de capacité définie par le service, la tâche autonome ou le cluster par défaut. Amazon ECS suit le reste de ces étapes pour un seul fournisseur de capacité.

Les tâches dépourvues de stratégie de fournisseur de capacité sont ignorées par les fournisseurs de capacité. Une tâche en attente qui n'est pas associée à une stratégie de fournisseur de capacité n'entraînera pas la montée en puissance d'un fournisseur de capacité. Les tâches ou les services ne peuvent pas définir de stratégie de fournisseur de capacité si elles définissent un type de lancement.

Le comportement de montée en puissance est décrit plus en détail ci-dessous.

- Regroupez toutes les tâches de provisionnement pour ce fournisseur de capacité de façon à ce chaque groupe ait exactement les mêmes besoins en ressources.
- Lorsque vous utilisez plusieurs types d'instances dans un groupe Auto Scaling, les types d'instances du groupe Auto Scaling sont triés par leurs paramètres. Ces paramètres incluent

vCPU, mémoire, interfaces réseau Elastic (ENI), ports et GPU. Les types d'instance les plus petits et les plus grands pour chaque paramètre sont sélectionnés. Pour plus d'informations sur la façon de choisir le type d'instances, consultez la section Instances de conteneur Amazon EC2 pour Amazon ECS.

> **Important**
>
> Si les besoins en ressources d'un groupe de tâches sont supérieurs à ceux du plus petit type d'instance du groupe Auto Scaling, ce groupe de tâches ne peut pas être exécuté avec ce fournisseur de capacité. Le fournisseur de capacité ne met pas le groupe Auto Scaling à l'échelle. Les tâches restent à l'état `PROVISIONING`.
> Pour éviter que les tâches ne restent à l'état `PROVISIONING`, nous vous recommandons de créer des groupes Auto Scaling et des fournisseurs de capacité distincts pour des besoins en ressources minimales différents. Lorsque vous exécutez des tâches ou créez des services, ajoutez uniquement des fournisseurs de capacité à la stratégie de fournisseur de capacité capables d'exécuter la tâche sur le plus petit type d'instance du groupe Auto Scaling. Pour les autres paramètres, vous pouvez utiliser des contraintes de placement.

- Pour chaque groupe de tâches, Amazon ECS calcule le nombre d'instances requises pour exécuter les tâches non placées. Ce calcul utilise une stratégie `binpack`. Cette stratégie tient compte des besoins en vCPU, mémoire, interfaces réseau Elastic (ENI), ports et GPU des tâches. Elle prend également en compte la disponibilité des ressources des instances Amazon EC2. Les valeurs des types d'instance les plus grands sont traitées comme le nombre maximal d'instances calculé. Les valeurs du plus petit type d'instance sont utilisées comme protection. Si le type d'instance le plus petit ne peut pas exécuter au moins une instance de la tâche, le calcul considère la tâche comme non compatible. Par conséquent, la tâche est exclue du calcul de montée en puissance. Lorsque toutes les tâches ne sont pas compatibles avec le plus petit type d'instance, l'autoscaling du cluster s'arrête et la valeur `CapacityProviderReservation` reste à `targetCapacity`.
- Amazon ECS publie le `CapacityProviderReservation` métrique à CloudWatch en ce qui concerne le `minimumScalingStepSize` si l'un des cas suivants est le cas.
  - Le nombre maximal d'instances calculé est inférieur à la taille minimale de l'étape de mise à l'échelle.
  - La plus faible des valeurs entre la `maximumScalingStepSize` et le nombre maximal d'instances calculé.

- CloudWatch les alarmes utilisent le `CapacityProviderReservation` métrique pour les fournisseurs de capacités. Lorsque la métrique `CapacityProviderReservation` est supérieure à la valeur `targetCapacity`, les alarmes augmentent également la `DesiredCapacity` du groupe Auto Scaling. Le `targetCapacity` la valeur est un paramètre du fournisseur de capacité envoyé au CloudWatch alarme pendant la phase d'activation de la mise à l'échelle automatique du cluster.

  La `targetCapacity` par défaut est de 100 %.

- Le groupe Auto Scaling lance des instances EC2 supplémentaires. Pour éviter la surallocation, l'autoscaling veille à ce que la capacité de l'instance EC2 récemment lancée soit stabilisée avant de lancer de nouvelles instances. La scalabilité automatique vérifie si toutes les instances existantes ont passé la `instanceWarmupPeriod` (maintenant moins l'heure de lancement de l'instance). L'augmentation horizontale est bloquée pour les instances en cours de `instanceWarmupPeriod`.

  Le nombre de secondes par défaut pour qu'une instance nouvellement lancée se prépare est de 300.

Pour plus d'informations, consultez Exploration approfondie de la mise à l'échelle automatique du cluster Amazon ECS.

Scale-out considérations

Tenez compte des points suivants pour le processus de montée en puissance :

- Bien qu'il existe plusieurs contraintes de placement, nous vous recommandons d'utiliser uniquement la contrainte de placement des tâches `distinctInstance`. Celle-ci empêche l'arrêt du processus de montée en puissance, car vous utilisez une contrainte de placement qui n'est pas compatible avec les instances échantillonnées.
- La mise à l'échelle gérée fonctionne mieux si votre groupe Auto Scaling utilise les mêmes types d'instance ou des types d'instance similaires.
- Lorsqu'un processus de mise à l'échelle horizontale est requis et qu'aucune instance de conteneur n'est en cours d'exécution, Amazon ECS monte toujours en puissance à deux instances dans un premier temps, puis exécute des processus de mise à l'échelle horizontale ou de montée en puissance supplémentaires. Toute montée en puissance attend la période de préparation de l'instance. Pour la mise à l'échelle horizontale, Amazon ECS attend 15 minutes après une montée en puissance avant de lancer des processus de mise à l'échelle horizontale à tout moment.

- La deuxième étape de montée en puissance doit attendre l'expiration du délai de la `instanceWarmupPeriod`, ce qui peut affecter la limite d'échelle globale. Si vous devez réduire ce délai, assurez-vous que la `instanceWarmupPeriod` est suffisamment importante pour que l'instance EC2 ait le temps de démarrer et lancer l'agent Amazon ECS (ce qui évite d'allouer trop de ressources).
- L'autoscaling de cluster prend en charge la configuration du lancement, les modèles de lancement et plusieurs types d'instances dans le groupe Auto Scaling du fournisseur de capacité. Vous pouvez également utiliser la sélection de type d'instance basée sur des attributs sans plusieurs types d'instances.
- Lorsque vous utilisez un groupe Auto Scaling avec On-Demand instances et types d'instances multiples ou instances ponctuelles, placez les types d'instances les plus importants plus haut dans la liste des priorités et ne spécifiez pas de pondération. La spécification d'une pondération n'est pas prise en charge pour le moment. Pour plus d'informations, consultez Groupes Auto Scaling avec types d'instance multiples dans le Guide de l'utilisateur AWS Auto Scaling .
- Amazon ECS lance ensuite la `minimumScalingStepSize`, si le nombre maximal d'instances calculé est inférieur à la taille minimale de l'étape de mise à l'échelle, ou la plus faible des valeurs entre `maximumScalingStepSize` et la valeur maximale calculée du nombre d'instances.
- Si un service Amazon ECS ou `run-task` lance une tâche et que les instances de conteneur du fournisseur de capacité ne disposent pas de ressources suffisantes pour démarrer la tâche, Amazon ECS limite le nombre de tâches ayant ce statut pour chaque cluster et empêche toute tâche de dépasser cette limite. Pour plus d'informations, consultez Service Quotas.

Comportement de mise à l'échelle horizontale gérée

Amazon ECS surveille les instances de conteneur pour chaque fournisseur de capacité au sein du cluster. Lorsqu'une instance de conteneur n'exécute aucune tâche, elle est considérée comme vide et Amazon ECS démarre le processus de mise à l'échelle horizontale.

CloudWatch les alarmes de mise à l'échelle nécessitent 15 points de données (15 minutes) avant que le processus d'intégration du groupe Auto Scaling ne commence. Une fois que le processus de mise à l'échelle horizontale démarre jusqu'à ce qu'Amazon ECS ait besoin de réduire le nombre d'instances de conteneur enregistrées, le groupe Auto Scaling définit la `DesireCapacity` pour qu'elle soit supérieure à une instance et inférieure à 50 % par minute.

Lorsqu'Amazon ECS demande une montée en puissance (lorsque la `CapacityProviderReservation` est supérieure à 100) alors qu'un processus de mise à l'échelle

horizontale est en cours, le processus de mise à l'échelle horizontale est arrêté et démarre dès le début si nécessaire.

Le comportement de mise à l'échelle horizontale est décrit plus en détail ci-dessous :

1. Amazon ECS calcule le nombre d'instances de conteneur vides. Une instance de conteneur est considérée comme vide même lorsque les tâches de démon sont exécutées.
2. Amazon ECS définit la valeur `CapacityProviderReservation` sur un nombre compris entre 0 et 100 qui utilise la formule suivante pour représenter le ratio entre la taille que doit avoir le groupe Auto Scaling par rapport à sa taille réelle, exprimé en pourcentage. Amazon ECS publie ensuite la métrique pour CloudWatch. Pour plus d'informations sur le calcul de la métrique, veuillez consulter le billet de blog Deep Dive on Amazon ECS Cluster Auto Scaling.

   ```
   CapacityProviderReservation = (number of instances needed) / (number of running
    instances) x 100
   ```

3. Le `CapacityProviderReservation` métrique génère un CloudWatch alarme. Cette alarme met à jour la valeur `DesiredCapacity` pour le groupe Auto Scaling. Ensuite, l'une des actions suivantes se produit :
   - Si vous n'utilisez pas la résiliation gérée par le fournisseur de capacité, le groupe Auto Scaling sélectionne les instances EC2 à l'aide de la politique de résiliation de groupe Auto Scaling et résilie les instances jusqu'à ce que le nombre d'instances EC2 atteigne la `DesiredCapacity`. L'enregistrement des instances de conteneur est ensuite annulé du cluster.
   - Si toutes les instances de conteneur utilisent la protection contre la résiliation gérée, Amazon ECS supprime la protection contre la mise à l'échelle horizontale sur les instances de conteneur qui sont vides. Le groupe Auto Scaling sera alors en mesure de résilier les instances EC2. L'enregistrement des instances de conteneur est ensuite annulé du cluster.

Contrôle des instances résiliées par Amazon ECS

> **Important**
>
> Vous devez activer la protection contre la mise à l'échelle horizontale d'instance Auto Scaling sur le groupe Auto Scaling pour utiliser la fonctionnalité de protection contre la résiliation gérée de l'autoscaling de cluster.

La protection contre la résiliation gérée permet à l'autoscaling de cluster de contrôler quelles instances sont résiliées. Lorsque vous utilisez la protection contre la résiliation gérée, Amazon ECS ne résilie que les instances EC2 qui n'ont pas de tâches Amazon ECS en cours d'exécution. Les tâches exécutées par un service utilisant la stratégie de planification `DAEMON` sont ignorées et une instance peut être résiliée par le biais de l'autoscaling de cluster, même lorsque l'instance exécute ces tâches. Cela est dû au fait que toutes les instances du cluster exécutent ces tâches.

Amazon ECS active d'abord l'option de protection d'instance contre la mise à l'échelle horizontale pour les instances EC2 du groupe Auto Scaling. Amazon ECS place ensuite les tâches sur les instances. Lorsque toutes les tâches autres que le démon sont arrêtées sur une instance, Amazon ECS lance le processus de mise à l'échelle horizontale et désactive la protection de la mise à l'échelle horizontale pour l'instance EC2. Le groupe Auto Scaling peut ensuite résilier l'instance.

La protection contre la mise à l'échelle horizontale d'instance Auto Scaling contrôle quelles instances EC2 peuvent être résiliées par Auto Scaling. Les instances dont la fonction de mise à l'échelle horizontale est activée ne peuvent pas être résiliées pendant le processus de mise à l'échelle horizontale. Pour plus d'informations sur la protection contre la mise à l'échelle horizontale d'instance Auto Scaling, veuillez consulter Utilisation de la protection contre la mise à l'échelle horizontale d'instance dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Vous pouvez définir le pourcentage de `targetCapacity` de manière à disposer d'une capacité de réserve. Cela permet de lancer plus rapidement les tâches futures, car le groupe Auto Scaling n'a pas besoin de lancer davantage d'instances. Amazon ECS utilise la valeur de capacité cible pour gérer CloudWatch métrique créée par le service. Amazon ECS gère le CloudWatch métrique. Le groupe Auto Scaling est traité comme un état stable de sorte qu'aucune action de mise à l'échelle ne soit requise. Les valeurs peuvent être comprises entre 0 et 100 %. Par exemple, pour configurer Amazon ECS de manière à conserver 10 % de capacité gratuite en plus de celle utilisée par les tâches Amazon ECS, définissez la valeur de capacité cible à 90 %. Tenez compte des points suivants lors de la définition de la valeur `targetCapacity` sur un fournisseur de capacité.

- Une valeur de `targetCapacity` inférieure à 100 % représente la quantité de capacité libre (instances Amazon EC2) devant être présente dans le cluster. La capacité libre signifie qu'il n'y a aucune tâche en cours d'exécution.
- Les contraintes de placement telles que les zones de disponibilité, sans `binpack` supplémentaire, forcent Amazon ECS à exécuter une tâche pour chaque instance, ce qui peut ne pas être le comportement souhaité.

Vous devez activer la protection contre la mise à l'échelle horizontale d'instance Auto Scaling sur le groupe Auto Scaling pour utiliser la protection contre la résiliation gérée. Si vous n'activez pas la protection contre la mise à l'échelle horizontale, l'activation de la protection contre la résiliation gérée peut entraîner un comportement indésirable. Par exemple, certaines instances peuvent être bloquées à l'état de drainage. Pour plus d'informations, consultez Protection contre la mise à l'échelle horizontale d'instance dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Lorsque vous utilisez la protection contre la résiliation auprès d'un fournisseur de capacité, n'effectuez aucune action manuelle, telle que le détachement de l'instance, sur le groupe Auto Scaling associé au fournisseur de capacité. Les actions manuelles peuvent interrompre la mise à l'échelle horizontale du fournisseur de capacité. Si vous détachez une instance du groupe Auto Scaling, vous devez également annuler l'enregistrement de l'instance détachée du cluster Amazon ECS.

Mise à jour de la protection contre la résiliation gérée pour les fournisseurs de capacité Amazon ECS

Lorsque vous utilisez la protection contre la résiliation gérée, vous devez mettre à jour les paramètres des fournisseurs de capacité existants.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page Cluster, choisissez l'onglet Infrastructure.
4. Choisissez le fournisseur de capacité.
5. Choisissez Mettre à jour pour modifier les paramètres du fournisseur de capacité.
6. Sous Paramètres du groupe Auto Scaling, activez ou désactivez la fonctionnalité Protection contre la résiliation gérée pour activer ou désactiver la fonctionnalité.
7. Choisissez Mettre à jour.

AWS CLI

Vous pouvez mettre à jour le paramètre de protection contre la résiliation gérée d'un fournisseur de capacité à l'aide de la commande `update-capacity-provider` suivante :

Pour activer la protection contre la résiliation gérée :

```
aws ecs update-capacity-provider \
```

```
  --name CapacityProviderName \
  --auto-scaling-group-provider
"managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

Pour désactiver la protection contre la résiliation gérée :

```
aws ecs update-capacity-provider \
  --name CapacityProviderName \
  --auto-scaling-group-provider
"managedScaling={status=ENABLED,targetCapacity=70,minimumScalingStepSize=1,maximumScalingStepS
```

> **Note**
>
> Les modifications peuvent prendre quelques minutes avant d'être effectives sur l'ensemble de votre cluster. Lorsque vous activez la protection contre la résiliation gérée, les instances qui exécutent déjà des tâches seront protégées contre les événements de réduction horizontale. Lorsque vous désactivez la protection contre la résiliation gérée, l'indicateur de protection sera supprimé des instances lors du prochain cycle de gestion du fournisseur de capacité ECS.

Console pour l'exécution de tâches

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page Cluster, choisissez l'onglet Tâches.
4. Choisissez la tâche.
5. Sous Configuration, basculez la Protection contre la résiliation gérée pour activer ou désactiver la fonctionnalité.
6. Choisissez Configurer la protection évolutive des tâches.

   La boîte de dialogue Configurer la protection intégrée des tâches s'affiche

   a. Sous Protection évolutive des tâches, basculer sur Activer.
   b. Pour Expire en minutes, saisissez le nombre de minutes avant la fin de la protection évolutive des tâches.
   c. Choisissez Update (Mettre à jour)

Activation de l'autoscaling de cluster Amazon ECS

Vous activez l'autoscaling de cluster de manière à ce qu'Amazon ECS gère la mise à l'échelle des instances Amazon EC2 enregistrées dans votre cluster.

Si vous souhaitez utiliser la console pour activer l'autoscaling de cluster, consultez la section Création d'un fournisseur de capacité pour Amazon ECS.

Avant de commencer, créez un groupe Auto Scaling et un fournisseur de capacité. Pour plus d'informations, consultez the section called "Fournisseurs de capacité Amazon ECS pour les charges de travail EC2".

Pour activer l'autoscaling de cluster, vous associez le fournisseur de capacité au cluster, puis vous activez l'autoscaling de cluster.

1. Utilisez la commande `put-cluster-capacity-providers` pour associer un ou plusieurs fournisseurs de capacité au cluster.

   Pour ajouter AWS Fargate les fournisseurs de capacités, notamment `FARGATE` et `FARGATE_SPOT` fournisseurs de capacités dans la demande. Pour plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName FARGATE FARGATE_SPOT \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

   Pour ajouter un groupe Auto Scaling pour EC2, incluez le nom du groupe Auto Scaling dans la requête. Pour plus d'informations, consultez `put-cluster-capacity-providers` dans la Référence des commandes de l'AWS CLI .

   ```
   aws ecs put-cluster-capacity-providers \
     --cluster ClusterName \
     --capacity-providers CapacityProviderName \
     --default-capacity-provider-strategy capacityProvider=CapacityProvider,weight=1
   ```

2. Utilisez la commande `describe-clusters` pour vérifier que l'association a réussi. Pour plus d'informations, consultez `describe-clusters` dans la Référence des commandes de l'AWS CLI .

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

3. Utilisez la commande `update-capacity-provider` pour activer la mise à l'échelle automatique gérée pour le fournisseur de capacité. Pour plus d'informations, consultez update-capacity-provider dans la Référence des commandes de l'AWS CLI .

```
aws ecs update-capacity-provider \
  --name CapacityProviderName \
  --auto-scaling-group-provider "managedScaling={status=ENABLED}"
```

Désactivation de l'autoscaling de cluster Amazon ECS

Vous désactivez l'autoscaling de cluster lorsque vous avez besoin d'un contrôle plus précis des instances EC2 enregistrées sur votre cluster,

Pour désactiver l'autoscaling de cluster, vous pouvez soit dissocier le fournisseur de capacité pour lequel la mise à l'échelle gérée du cluster est activée, soit mettre à jour le fournisseur de capacité afin de désactiver la mise à l'échelle gérée.

Dissociation du fournisseur de capacité

Effectuez les étapes suivantes pour dissocier le fournisseur de capacité d'un cluster.

1. Utilisez la commande `put-cluster-capacity-providers` pour dissocier le fournisseur de capacité de groupe Auto Scaling du cluster. Le cluster peut conserver l'association avec AWS Fargate fournisseurs de capacités. Pour plus d'informations, consultez put-cluster-capacity-providers dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers FARGATE FARGATE_SPOT \
  --default-capacity-provider-strategy '[]'
```

Utilisez la commande `put-cluster-capacity-providers` pour dissocier le fournisseur de capacité de groupe Auto Scaling du cluster. Pour plus d'informations, consultez put-cluster-capacity-providers dans la Référence des commandes de l'AWS CLI .

```
aws ecs put-cluster-capacity-providers \
  --cluster ClusterName \
  --capacity-providers [] \
  --default-capacity-provider-strategy '[]'
```

2. Utilisez la commande `describe-clusters` pour vérifier que la dissociation a réussi. Pour plus d'informations, consultez la section `describe-clusters` dans la référence des commandes AWS CLI .

```
aws ecs describe-clusters \
  --cluster ClusterName \
  --include ATTACHMENTS
```

Désactivez la mise à l'échelle gérée pour le fournisseur de capacité

Effectuez les étapes suivantes pour désactiver la mise à l'échelle gérée pour le fournisseur de capacité.

- Utilisez la commande `update-capacity-provider` pour désactiver la mise à l'échelle automatique gérée pour le fournisseur de capacité. Pour plus d'informations, consultez `update-capacity-provider` dans la Référence des commandes de l'AWS CLI .

```
aws ecs update-capacity-provider \
  --name CapacityProviderName \
  --auto-scaling-group-provider "managedScaling={status=DISABLED}"
```

Création d'un fournisseur de capacité pour Amazon ECS

Une fois la création du cluster terminée, vous pouvez créer un fournisseur de capacité (groupe Auto Scaling) pour EC2. Les fournisseurs de capacité vous aident à gérer et à mettre à l'échelle votre infrastructure pour vos applications.

Avant de créer le fournisseur de capacité, vous devez créer un groupe Auto Scaling. Pour plus d'informations, voir Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Pour créer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures, puis choisissez Créez.
5. Sur la page Create capacity provider (Créer des fournisseurs de capacité), configurez les options suivantes.

    a. Sous Basic details (Détails de base), pour le Capacity provider name (Nom du fournisseur de capacité), entrez un nom de fournisseur de capacité.

    b. Sous Auto Scaling group (Groupe Auto Scaling), pour Use an existing Auto Scaling group (Utiliser un groupe Auto Scaling existant), choisissez le groupe Auto Scaling.

    c. (Facultatif) Pour configurer une stratégie de dimensionnement, sous Stratégies de dimensionnement, configurez les options suivantes.

    - Pour qu'Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance, sélectionnez Turn on managed scaling (Activer la mise à l'échelle gérée).
    - Pour empêcher la résiliation de l'instance EC2 exécutant des tâches Amazon ECS, sélectionnez Turn on scaling protection (Activer la protection du dimensionnement).
    - Pour Définir la capacité cible, entrez la valeur cible pour CloudWatch métrique utilisée en Amazonie ECS-managed politique de mise à l'échelle du suivi des cibles.

6. Cliquez sur Créer.

Mise à jour d’un fournisseur de capacité Amazon ECS

Lorsque vous utilisez un groupe Auto Scaling comme fournisseur de capacité, vous pouvez modifier la stratégie de dimensionnement du groupe.

Pour mettre à jour un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures, puis choisissez Mettre à jour.
5. Sur la page Create capacity provider (Créer des fournisseurs de capacité), configurez les options suivantes.

- Sous Groupe Auto Scaling de Scaling policies (Stratégies de dimensionnement), configurez les options suivantes.
  - Pour qu'Amazon ECS gère les actions de mise à l'échelle horizontale et de montée en puissance, sélectionnez Turn on managed scaling (Activer la mise à l'échelle gérée).
  - Pour empêcher la résiliation des instances EC2 exécutant des tâches Amazon ECS, sélectionnez Activer la protection du dimensionnement.
  - Pour Définir la capacité cible, entrez la valeur cible pour CloudWatch métrique utilisée en Amazonie ECS-managed politique de mise à l'échelle du suivi des cibles.

6. Choisissez Mettre à jour.

Suppression d’un fournisseur de capacité Amazon ECS

Si vous avez fini d'utiliser un fournisseur de capacité de groupe Auto Scaling, vous pouvez le supprimer. Une fois le groupe supprimé, le fournisseur de capacité de groupe Auto Scaling passe à l’état `INACTIVE`. Les fournisseurs de capacité ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des fournisseurs de capacité dont l'état est `INACTIVE`. Avant que le fournisseur de capacité de groupe Auto Scaling ne soit supprimé, il doit être retiré de la stratégie de fournisseur de capacité de tous les services. Vous pouvez utiliser l'API `UpdateService` ou le flux de service de mise à jour dans la console Amazon ECS pour supprimer un fournisseur de capacité de la stratégie de fournisseur de capacité d'un service. Utilisez l’option Forcer le nouveau déploiement pour vous assurer que toutes les tâches utilisant la capacité d’instance Amazon EC2 fournie par le fournisseur de capacité sont basculées vers l’utilisation de la capacité des autres fournisseurs de capacité.

Pour supprimer un fournisseur de capacité pour le cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures, le groupe Auto Scaling, puis choisissez Supprimer.
5. Dans la case de confirmation, entrez supprimer ***Auto Scaling group name***
6. Sélectionnez Supprimer.

Arrêt en toute sécurité des charges de travail Amazon ECS exécutées sur les instances EC2

Le drainage d'instance gérée facilite la résiliation progressive des instances Amazon EC2. Cela permet à vos charges de travail de s'arrêter en toute sécurité et d'être reprogrammées vers des instances qui ne sont pas en cours de résiliation. La maintenance et les mises à jour de l'infrastructure sont effectuées sans perturber les charges de travail. En utilisant le drainage d'instance gérée, vous simplifiez les flux de travail de gestion de votre infrastructure qui nécessitent le remplacement des instances Amazon EC2, tout en garantissant la résilience et la disponibilité de vos applications.

Le drainage d'instance gérée Amazon ECS fonctionne avec les remplacements d'instances de groupe Auto Scaling. Sur la base de l'actualisation des instances et de leur durée de vie maximale, les clients peuvent s'assurer qu'ils restent conformes aux dernières réglementations en matière de système d'exploitation et de sécurité en matière de capacité.

Le drainage d'instance gérée ne peut être utilisé qu'avec les fournisseurs de capacité Amazon ECS. Vous pouvez activer le drainage géré des instances lorsque vous créez ou mettez à jour vos fournisseurs de capacité de groupe Auto Scaling à l'aide de la console Amazon ECS, AWS CLI, ou SDK.

Les événements suivants sont couverts par le drainage d'instance gérée Amazon ECS.

- Actualisation de l'instance du groupe Auto Scaling : utilisez l'actualisation des instances pour effectuer le remplacement progressif de vos instances Amazon EC2 dans votre groupe Auto Scaling au lieu de le faire manuellement par lots. Cette méthode est très utile lorsque vous devez remplacer un grand nombre d'instances. Une actualisation d'instance est lancée via la console Amazon EC2 ou l'API `StartInstanceRefresh`. Assurez-vous de sélectionner `Replace` pour Scale-in protection lors des appels `StartInstanceRefresh` si vous utilisez une protection gérée contre les interruptions de service.
- Durée de vie maximale des instances : vous pouvez définir une durée de vie maximale lorsqu'il s'agit de remplacer les instances du groupe Auto Scaling. Cette méthode est utile pour planifier le remplacement des instances en fonction des politiques de sécurité internes ou de la conformité.
- Mise à l'échelle du groupe Auto Scaling : sur la base des politiques de mise à l'échelle et des actions de mise à l'échelle planifiées, le groupe Auto Scaling prend en charge la mise à l'échelle automatique des instances. En utilisant un groupe Auto Scaling comme fournisseur de capacité Amazon ECS, vous pouvez intégrer des instances de groupe Auto Scaling lorsqu'aucune tâche n'y est exécutée.

- Surveillances de l'état du groupe Auto Scaling : le groupe Auto Scaling prend en charge de nombreuses surveillances de l'état pour gérer la résiliation d'instances défectueuses.
- CloudFormation mises à jour des piles - Vous pouvez ajouter `UpdatePolicy` attribuer à votre CloudFormation stack pour effectuer des mises à jour continues lorsque le groupe change.
- Rééquilibrage de la capacité Spot : le groupe Auto Scaling essaie de remplacer de manière proactive les instances Spot présentant un risque d'interruption plus élevé sur la base de l'évaluation du rééquilibrage de capacité Amazon EC2. Le groupe Auto Scaling résilie l'ancienne instance lorsque la solution de remplacement est lancée et saine. Le drainage d'instance gérée Amazon ECS draine l'instance Spot de la même manière qu'une instance non Spot.
- Interruption Spot : les instances Spot sont résiliées avec un préavis de deux minutes. Amazon ECS-managed Le drainage d'instance place l'instance en état de vidange en réponse.

Hooks de cycle de vie Amazon EC2 Auto Scaling avec drainage d'instance gérée

Les hooks du cycle de vie du groupe Auto Scaling permettent au client de créer des solutions déclenchées par certains événements du cycle de vie de l'instance et d'effectuer une action personnalisée lorsque cet événement se produit. Un groupe Auto Scaling autorise jusqu'à 50 hooks. Plusieurs hooks de résiliation peuvent exister et être exécutés en parallèle. Le groupe Auto Scaling attend que tous les hooks soient terminés avant de résilier une instance.

En plus de l'Amazon ECS-managed terminaison des crochets, vous pouvez également configurer vos propres crochets de terminaison du cycle de vie. Les hooks du cycle de vie ont une `default action`, et nous vous recommandons de le définir sur `continue` par défaut pour garantir que les autres hooks, tels que le hook géré par Amazon ECS, ne soient pas affectés par les erreurs provenant des hooks personnalisés.

Si vous avez déjà configuré un hook de fin de cycle de vie de groupe Auto Scaling et que vous avez également activé le drainage d'instance gérée Amazon ECS, les deux hooks de cycle de vie sont exécutés. Les horaires relatifs ne sont toutefois pas garantis. Les hooks du cycle de vie disposent d'un paramètre `default action` permettant de spécifier l'action à entreprendre lorsque le délai expire. En cas d'échec, nous vous recommandons d'utiliser `continue` comme résultat par défaut votre hook personnalisé. Cela garantit que les autres hooks, en particulier les hooks gérés par Amazon ECS, ne sont pas affectés par des erreurs dans votre hook de cycle de vie personnalisé. Le résultat alternatif `abandon` provoque l'ignorance de tous les autres hooks et doit être évité. Pour plus d'informations sur les hooks du cycle de vie des groupes Auto Scaling, consultez la section Hooks du cycle de vie Amazon EC2 Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Tâches et drainage d'instance gérée

Le drainage d'instance gérée Amazon ECS utilise la fonctionnalité de drainage existante présente dans les instances de conteneur. La fonctionnalité de drainage des instances de conteneur remplace et arrête les tâches de réplica appartenant à un service Amazon ECS. Une tâche autonome, telle qu'une tâche invoquée par `RunTask`, qui est à l'état `PENDING` ou `RUNNING` reste inchangée. Vous devez attendre qu'elles soient terminées ou qu'elles soient arrêtées manuellement. L'instance de conteneur reste dans l'état `DRAINING` jusqu'à ce que toutes les tâches soient arrêtées ou jusqu'à ce que 48 heures se soient écoulées. Les tâches du démon sont les dernières à s'arrêter après l'arrêt de toutes les tâches de réplica.

Drainage d'instance gérée et protection contre la résiliation gérée

Le drainage d'instance gérée fonctionne même si la résiliation gérée est désactivée. Pour plus d'informations sur la protection contre la résiliation gérée, consultez la section Contrôle des instances résiliées par Amazon ECS.

Le tableau suivant récapitule le comportement des différentes combinaisons de résiliation gérée et de drainage géré.

Résiliation gérée	Drainage géré	Outcome
Activé	Activé	Amazon ECS empêche les instances Amazon EC2 qui exécutent des tâches d'être résiliées par des événements de réduction horizontale. Toutes

Résiliation gérée	Drainage géré	Outcome
		instances en cours de résiliation, telles que celles pour lesquelles la protection contre la résiliation n'est pas définie, celles qui font l'objet d'une interruption Spot ou qui sont forcées par une actualisation de l'instance, sont drainées progressivement.

Résiliation gérée	Drainage géré	Outcome
Désactivé	Activé	Amazon ECS ne protège pas les instances Amazon EC2 exécutant des tâches contre la réduction horizontale. Cependant, toutes les instances résiliées sont drainées progressivement.

Résiliation gérée	Drainage géré	Outcome
Activé	Désactivé	Amazon ECS empêche les instances Amazon EC2 qui exécutent des tâches d'être résiliées par des événements de réduction horizontale. Cependant, les instances peuvent toujours être résiliées en cas d'interruption Spot ou d'actualisation forcée de l'instance, ou si elles n'exécutent aucune tâche.

Résiliation gérée	Drainage géré	Outcome
		Amazon ECS n'effectue pas de drainage progressif pour ces instances et lance des tâches de service de remplacement après leur arrêt.

Résiliation gérée	Drainage géré	Outcome
Désactivé	Désactivé	Les instances Amazon EC2 peuvent faire l'objet d'une réduction horizonta le ou être résiliées à tout moment, même si elles exécutent des tâches Amazon ECS. Amazon ECS lancera les tâches de service de remplacem ent après leur arrêt.

Drainage d'instance gérée et drainage d'instance Spot

Avec le drainage d'instance Spot, vous pouvez définir une variable d'environnement `ECS_ENABLE_SPOT_INSTANCE_DRAINING` sur l'agent Amazon ECS permettant à Amazon ECS de placer une instance en état de drainage en réponse à une interruption Spot de deux minutes. Le drainage d'instance gérée Amazon ECS facilite l'arrêt progressif des instances Amazon EC2 en cours de résiliation pour de nombreuses raisons, et pas uniquement pour une interruption Spot. Par exemple, vous pouvez utiliser le rééquilibrage de capacité d'Amazon EC2 Auto Scaling pour remplacer de manière proactive une instance Spot en cas de risque élevé d'interruption, et le

drainage d'instance gérée permet d'arrêter progressivement l'instance Spot remplacée. Lorsque vous utilisez le drainage d'instance gérée, il n'est pas nécessaire d'activer le drainage d'instance Spot séparément. Par conséquent, la spécification de `ECS_ENABLE_SPOT_INSTANCE_DRAINING` dans les données utilisateur du groupe Auto Scaling est redondant. Pour plus d'informations sur les instances Spot, consultez la section Instances Spot.

Comment fonctionne le drainage géré des instances avec EventBridge

Les événements de drainage des instances gérées par Amazon ECS sont publiés sur Amazon EventBridge, et Amazon ECS crée un EventBridge règle gérée dans le bus par défaut de votre compte pour prendre en charge le drainage géré des instances. Vous pouvez filtrer ces événements selon d'autres AWS des services tels que Lambda, Amazon SNS et Amazon SQS pour surveiller et résoudre les problèmes.

- Amazon EC2 Auto Scaling envoie un événement à EventBridge lorsqu'un hook de cycle de vie est invoqué.
- Les avis d'interruption ponctuelle sont publiés à EventBridge.
- Amazon ECS génère des messages d'erreur que vous pouvez récupérer via la console et les API Amazon ECS.
- EventBridge dispose de mécanismes de nouvelle tentative intégrés pour atténuer les défaillances temporaires.

Configuration des fournisseurs de capacité Amazon ECS pour l'arrêt des instances en toute sécurité

Vous pouvez activer le drainage d'instance gérée lorsque vous créez ou mettez à jour les fournisseurs de capacité de votre groupe Auto Scaling à l'aide de la console et AWS CLI Amazon ECS.

> **Note**
>
> Le drainage d'instance gérée est activé par défaut lorsque vous créez un fournisseur de capacité.

Vous trouverez ci-dessous des exemples utilisant AWS CLI pour créer un fournisseur de capacité avec le drainage d'instance géré activé et activer le drainage d'instance géré pour le fournisseur de capacité existant d'un cluster.

Création d'un fournisseur de capacité avec le drainage d'instance gérée

Pour créer un fournisseur de capacité avec le drainage d'instance gérée activé, utilisez la commande `create-capacity-provider`. Définissez le paramètre `managedDraining` sur `ENABLED`.

```
aws ecs create-capacity-provider \
--name capacity-provider \
--auto-scaling-group-provider '{
  "autoScalingGroupArn": "asg-arn",
  "managedScaling": {
    "status": "ENABLED",
    "targetCapacity": 100,
    "minimumScalingStepSize": 1,
    "maximumScalingStepSize": 1
  },
  "managedDraining": "ENABLED",
  "managedTerminationProtection": "ENABLED",
}'
```

Réponse :

```
{
    "capacityProvider": {
        "capacityProviderArn": "capacity-provider-arn",
        "name": "capacity-provider",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1
            },
            "managedTerminationProtection": "ENABLED"
            "managedDraining": "ENABLED"
        }
    }
}
```

Activation du drainage d'instance gérée pour le fournisseur de capacité existant d'un cluster

Activez le drainage d'instance gérée pour le fournisseur de capacité existant d'un cluster à l'aide de la commande `update-capacity-provider`. Vous voyez que `managedDraining` a pour valeur actuelle `DISABLED` et que `updateStatus` a pour valeur actuelle `UPDATE_IN_PROGRESS`.

```
aws ecs update-capacity-provider \
--name cp-draining \
--auto-scaling-group-provider '{
  "managedDraining": "ENABLED"
}
```

Réponse :

```
{
    "capacityProvider": {
        "capacityProviderArn": "cp-draining-arn",
        "name": "cp-draining",
        "status": "ACTIVE",
        "autoScalingGroupProvider": {
            "autoScalingGroupArn": "asg-draining-arn",
            "managedScaling": {
                "status": "ENABLED",
                "targetCapacity": 100,
                "minimumScalingStepSize": 1,
                "maximumScalingStepSize": 1,
                "instanceWarmupPeriod": 300
            },
            "managedTerminationProtection": "DISABLED",
            "managedDraining": "DISABLED" // before update
        },
        "updateStatus": "UPDATE_IN_PROGRESS", // in progress and need describe again to
 find out the result
        "tags": [
        ]
    }
}
```

Utilisez la commande `describe-clusters` et incluez l'option `ATTACHMENTS`. Le `status` de l'attachement du drainage d'instance gérée est `PRECREATED`, et le `attachmentsStatus` global est `UPDATING`.

```
aws ecs describe-clusters --clusters cluster-name --include ATTACHMENTS
```

Réponse :

```
{
    "clusters": [
        {
            ...

            "capacityProviders": [
                "cp-draining"
            ],
            "defaultCapacityProviderStrategy": [],
            "attachments": [
                # new precreated managed draining attachment
                {
                    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                    "type": "managed_draining",
                    "status": "PRECREATED",
                    "details": [
                        {
                            "name": "capacityProviderName",
                            "value": "cp-draining"
                        },
                        {
                            "name": "autoScalingLifecycleHookName",
                            "value": "ecs-managed-draining-termination-hook"
                        }
                    ]
                },

                ...

            ],
            "attachmentsStatus": "UPDATING"
        }
    ],
    "failures": []
}
```

Lorsque la mise à jour est terminée, utilisez `describe-capacity-providers`, et vous verrez que `managedDraining` a maintenant pour valeur `ENABLED`.

```
aws ecs describe-capacity-providers --capacity-providers cp-draining
```

Réponse :

```
{
    "capacityProviders": [
        {
            "capacityProviderArn": "cp-draining-arn",
            "name": "cp-draining",
            "status": "ACTIVE",
            "autoScalingGroupProvider": {
                "autoScalingGroupArn": "asg-draning-arn",
                "managedScaling": {
                    "status": "ENABLED",
                    "targetCapacity": 100,
                    "minimumScalingStepSize": 1,
                    "maximumScalingStepSize": 1,
                    "instanceWarmupPeriod": 300
                },
                "managedTerminationProtection": "DISABLED",
                "managedDraining": "ENABLED" // successfully update
            },
            "updateStatus": "UPDATE_COMPLETE",
            "tags": []
        }
    ]
}
```

Résolution des problèmes liés au drainage d'instance gérée Amazon ECS

Il se peut que vous deviez résoudre les problèmes liés au drainage d'instance gérée. Vous trouverez ci-dessous un exemple de problème et de résolution que vous pourriez rencontrer lors de son utilisation.

Les instances ne s'arrêtent pas après avoir dépassé leur durée de vie maximale lors de l'utilisation de l'autoscaling.

Si vos instances ne s'arrêtent pas même après avoir atteint ou dépassé la durée de vie maximale lors de l'utilisation d'un groupe Auto Scaling, c'est peut-être parce qu'elles sont protégées contre la réduction horizontale. Vous pouvez désactiver la gestion de la résiliation et autoriser le drainage géré pour gérer le recyclage des instances.

Comportement du drainage pour les instances gérées Amazon ECS

La résiliation d'Amazon ECS Managed Instances garantit des transitions de charge de travail harmonieuses tout en optimisant les coûts et en préservant l'état de santé du système. Le système de résiliation offre trois voies décisionnelles distinctes pour la résiliation d'instance, chacune présentant des caractéristiques différentes en termes de délais et d'impact sur le client.

Voies de décision en matière de résiliation

Customer-initiated résiliation

Permet de contrôler directement la suppression des instances lorsque vous devez immédiatement retirer des instances de conteneur du service. Vous invoquez le DeregisterContainerInstance API dont l'indicateur de force est défini sur true, indiquant qu'une résiliation immédiate est requise malgré toutes les charges de travail en cours.

System-initiated terminaison inactive

Les instances gérées Amazon ECS surveillent en permanence et optimisent les coûts de manière proactive en mettant fin aux instances de conteneur Amazon ECS inactives qui n'exécutent aucune tâche. ECS utilise un délai heuristique pour donner aux instances de conteneurs la possibilité d'acquérir les tâches récemment lancées avant d'être interrompues. Cela peut être personnalisé avec `scaleInAfter` Paramètre de configuration du fournisseur de capacité Amazon ECS Managed Instances.

Résiliation provoquée par l'actualisation de l'infrastructure

Amazon ECS Managed Instances gère et met à jour automatiquement les logiciels des instances de conteneurs gérées afin de garantir la sécurité et la conformité tout en maintenant la disponibilité de la charge de travail. Pour plus d'informations, voir application de correctifs dans les instances gérées Amazon ECS.

Drainage progressif et migration de la charge de travail

Le système de drainage progressif met en œuvre une coordination sophistiquée avec la gestion des services Amazon ECS afin de garantir que les tâches gérées par les services sont correctement migrées hors des instances dont l'arrêt est prévu.

Coordination du drainage des tâches de service

Lorsqu'une instance passe à l'état DRAINAGE, le planificateur Amazon ECS arrête automatiquement de placer de nouvelles tâches sur l'instance tout en mettant en œuvre des procédures d'arrêt

progressif pour les tâches de service existantes. Le drainage des tâches de service inclut la coordination avec les stratégies de déploiement de service, les exigences en matière de surveillance de l'état et vos préférences en matière de drainage, afin de garantir un calendrier de migration et des taux de réussite optimaux.

Gestion autonome des tâches

Les tâches autonomes nécessitent un traitement différent, car elles ne bénéficient pas de la gestion automatique des services. Le système évalue les caractéristiques des tâches autonomes, notamment les estimations de durée des tâches, l'analyse des probabilités d'achèvement et l'évaluation de l'impact sur le client. La stratégie d'exécution progressive permet aux tâches autonomes de s'exécuter naturellement pendant une période de grâce prolongée, tandis que la résiliation forcée garantit que l'actualisation de l'infrastructure se produit dans des délais acceptables lorsque les tâches ne se sont pas résiliées naturellement.

Two-phase stratégie d'achèvement

Le système de résiliation met en œuvre une approche en deux phases qui équilibre la continuité de la charge de travail avec les exigences de gestion de l'infrastructure.

Phase 1 : période d'achèvement progressif

Au cours de cette phase, le système met en œuvre des stratégies de drainage progressif qui privilégient la continuité de la charge de travail. Les tâches de service sont progressivement drainées par le biais des processus de planification habituels d'Amazon ECS, les tâches autonomes continuent de s'exécuter et peuvent se terminer naturellement, et le système surveille que toutes les tâches atteignent l'état d'arrêt grâce à des processus d'achèvement naturels.

Phase 2 : application stricte des délais

Lorsque l'achèvement progressif ne permet pas d'atteindre les objectifs de résiliation dans des délais acceptables, le système met en œuvre une application stricte des délais. Le délai strict est généralement fixé à la date de début du drainage plus sept jours, ce qui laisse suffisamment de temps pour mener à bien l'opération tout en respectant les exigences opérationnelles. L'application de ce délai comprend l'invocation automatique des procédures de désinscription forcée et l'arrêt immédiat de toutes les tâches restantes, quel que soit le statut d'achèvement.

Création de ressources pour la mise à l'échelle automatique du cluster Amazon ECS à l'aide du Console de gestion AWS

Découvrez comment créer les ressources pour l’autoscaling de cluster à l’aide de l’ Console de gestion AWS. Lorsque les ressources nécessitent un nom, nous utilisons le préfixe `ConsoleTutorial` afin de garantir qu’elles ont toutes un nom unique et de faciliter leur localisation.

Rubriques

- Conditions préalables
- Étape 1 : Créer un cluster Amazon ECS
- Étape 2 : Enregistrer une définition de tâche
- Étape 3 : Exécuter une tâche
- Étape 4 : Vérifier
- Étape 5 : nettoyer

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l’exemple de politique IAM Amazon ECS_FullAccess.
- Le rôle IAM d'instance de conteneur Amazon ECS est créé. Pour plus d’informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
- Le rôle IAM lié à un service Amazon ECS service est créé. Pour plus d’informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- Le rôle IAM lié à un service Auto Scaling est créé. Pour plus d'informations, voir Service-Linked Rôles pour Amazon EC2 Auto Scaling dans le Guide de l'utilisateur d'Amazon EC2 Auto Scaling.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Pour plus d’informations, consultez the section called “Créer un Virtual Private Cloud”.

Étape 1 : Créer un cluster Amazon ECS

Pour créer un cluster Amazon ECS, effectuez les étapes suivantes.

Amazon ECS crée un modèle de lancement Amazon EC2 Auto Scaling et un groupe Auto Scaling en votre nom dans le cadre du CloudFormation empiler.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, sélectionnez Clusters, puis Créer un cluster.
3. Sous Cluster configuration (Configuration de cluster), pour Cluster name (Nom du cluster), saisissez `ConsoleTutorial-cluster`.
4. En dessous Infrastructures, clair AWS Fargate (sans serveur), puis sélectionnez Instances Amazon EC2. Ensuite, configurez le groupe Auto Scaling qui agit en tant que fournisseur de capacité.
   - Dans Groupe Auto Scaling (ASG). Sélectionnez Créer un ASG, puis fournissez les informations suivantes sur le groupe :
     - Pour Fonctionnement system/Architecture, choisissez Amazon Linux 2.
     - Pour EC2 instance type (Type d'instance EC2), choisissez t3.nano.
     - Pour Capacity (Capacité), saisissez le nombre minimum et le nombre maximum d'instances à lancer dans le groupe Auto Scaling.
5. (Facultatif) Pour gérer les identifications de cluster, développez Tags (Identifications), puis effectuez l'une des opérations suivantes :

   [Add a tag] Choisissez Add tag (Ajouter une étiquette) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Value (Valeur), saisissez la valeur de clé.

   [Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de l'étiquette.
6. Cliquez sur Créer.

Étape 2 : Enregistrer une définition de tâche

Avant de pouvoir exécuter une tâche sur votre cluster, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui utilise une image `amazonlinux` de Docker Hub et qui est simplement en veille. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, collez le contenu suivant.

```
{
    "family": "ConsoleTutorial-taskdef",
    "containerDefinitions": [
        {
            "name": "sleep",
            "image": "public.ecr.aws/amazonlinux/amazonlinux:latest",
            "memory": 20,
            "essential": true,
            "command": [
                "sh",
                "-c",
                "sleep infinity"
            ]
        }
    ],
    "requiresCompatibilities": [
        "EC2"
    ]
}
```

5. Cliquez sur Créer.

Étape 3 : Exécuter une tâche

Après avoir enregistré une définition de tâche pour votre compte, vous pouvez exécuter une tâche dans le cluster. Pour ce didacticiel, vous exécutez cinq instances de la définition de tâche `ConsoleTutorial-taskdef` de votre cluster `ConsoleTutorial-cluster`.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur le Clusters page, choisissez ConsoleTutorial-cluster.
3. Dans Tâches, choisissez Exécuter une nouvelle tâche.
4. Dans la section Environnement, sous Options de calcul, choisissez Stratégie de fournisseur de capacité.

5. Sous Configuration de déploiement, pour Type d'application, sélectionnez Tâche.
6. Choisissez ConsoleTutorial-taskdef à partir du Famille liste déroulante.
7. Sous Tâches souhaitées, saisissez 5.
8. Cliquez sur Créer.

Étape 4 : Vérifier

À ce stade du didacticiel, vous devez avoir un cluster avec cinq tâches en cours d'exécution et un groupe Auto Scaling avec un fournisseur de capacité. La mise à l'échelle gérée Amazon ECS est activée pour le fournisseur de capacité.

Nous pouvons vérifier que tout fonctionne correctement en consultant le CloudWatch les métriques, les paramètres du groupe Auto Scaling et enfin le nombre de tâches du cluster Amazon ECS.

Pour consulter le CloudWatch statistiques pour votre cluster

1. Ouvrez le CloudWatch console à https://console.aws.amazon.com/cloudwatch/.
2. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
3. Dans le panneau de navigation, sous Métriques, sélectionnez Toutes les métriques.
4. Sur la page Toutes les métriques, sous l'onglet Parcourir, sélectionnez `AWS/ECS/ManagedScaling`.
5. Choisissez CapacityProviderName, ClusterName.
6. Cochez la case qui correspond à `ConsoleTutorial-cluster` ClusterName.
7. Dans l'onglet Graphique des métriques, faites passer la valeur de Période à 30 secondes et celle de Statistique à Maximum.

   La valeur affichée dans le graphique indique la valeur de capacité cible pour le fournisseur de capacité. Elle doit commencer à `100`, ce qui correspond au pourcentage de capacité cible que nous avons défini. Elle doit ensuite monter à `200`, ce qui déclenche une alarme en vertu de la politique de suivi des objectifs et d'échelonnement. L'alarme déclenche alors la montée en puissance du groupe Auto Scaling.

Effectuez les étapes suivantes pour afficher les détails de votre groupe Auto Scaling et vérifier que la montée en puissance a bien eu lieu.

Pour vérifier que le groupe Auto Scaling est monté en puissance

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
3. Dans le volet de navigation, sous Auto Scaling, choisissez Auto Scaling Groups (Groupes Auto Scaling).
4. Choisissez le groupe Auto Scaling `ConsoleTutorial-cluster` créé dans ce didacticiel. Consultez la valeur sous Capacité souhaitée et consultez les instances sous l'onglet Gestion des instances pour confirmer que votre groupe est monté en puissance de deux instances.

Effectuez les étapes suivantes pour examiner votre cluster Amazon ECS et vérifier que les instances Amazon EC2 ont bien été enregistrées auprès du cluster et que vos tâches sont passées à l'état `RUNNING`.

Vérifier les instances dans le groupe Auto Scaling

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster `ConsoleTutorial-cluster`.
4. Dans l'onglet Tâches, vérifiez qu'il y a bien cinq tâches à l'état `RUNNING`.

Étape 5 : nettoyer

Lorsque vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas. La suppression des fournisseurs de capacité et des définitions de tâches n'est pas prise en charge, mais ces ressources n'entraînent aucun coût.

Pour nettoyer les ressources du didacticiel

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur le Clusters page, choisissez ConsoleTutorial-cluster.
4. Sur le ConsoleTutorial-cluster page, choisissez Tâches onglet, puis choisissez Arrête, Tout arrêter.
5. Dans le panneau de navigation, choisissez Clusters.

6. Sur le Clusters page, choisissez ConsoleTutorial-cluster.
7. Choisissez Supprimer le cluster.
8. Dans la case de confirmation, entrez supprimer ConsoleTutorial-cluster et choisissez Supprimer.
9. Supprimez les groupes Auto Scaling en effectuant les étapes suivantes.

    a. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
    b. Dans la barre de navigation, en haut de l'écran, sélectionnez la région .
    c. Dans le volet de navigation, sous Auto Scaling, choisissez Auto Scaling Groups (Groupes Auto Scaling).
    d. Sélectionnez le groupe Auto Scaling `ConsoleTutorial-cluster`, puis sélectionnez Actions.
    e. Dans le menu Actions, choisissez Delete (Supprimer). Saisissez delete dans la zone de confirmation, puis choisissez Supprimer.

Instances de conteneur Amazon EC2 pour Amazon ECS

Une instance de conteneur Amazon ECS est une instance Amazon EC2 qui exécute l'agent de conteneur Amazon ECS et qui est enregistrée dans un cluster. Lorsque vous exécutez des tâches avec Amazon ECS à l'aide du fournisseur de capacité, du fournisseur de capacité externe ou d'un fournisseur de capacité de groupe Auto Scaling, vos tâches sont placées sur vos instances de conteneur actives. Vous êtes responsable de la gestion et de la maintenance des instances de conteneur.

Bien que vous puissiez créer votre propre AMI d'instance Amazon EC2 répondant aux spécifications de base requises pour exécuter vos charges de travail conteneurisées sur Amazon ECS, Amazon ECS-optimized Les AMI sont préconfigurées et testées sur Amazon ECS par AWS ingénieurs. C'est la façon la plus simple de démarrer et d'exécuter rapidement vos conteneurs sur AWS .

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Quand vous utilisez CloudFormation pour créer un cluster, le paramètre SSM fait partie du modèle de lancement Amazon EC2 pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée pour Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, une vérification est effectuée pour voir s'il existe une mise à jour disponible qui doit être appliquée aux instances EC2. Pour un exemple d'utilisation du paramètre Systems Manager, voir

Créez un cluster Amazon ECS avec Amazon ECS-optimized AMI Amazon Linux 2023 dans le AWS CloudFormation Guide de l'utilisateur.

- Récupération d'Amazon ECS-optimized Métadonnées AMI Linux
- Récupération d'Amazon ECS-optimized Fusée à bouteilles Métadonnées AMI
- Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows

Vous pouvez choisir parmi les types d'instances compatibles avec votre application. Avec des instances plus grandes, vous pouvez lancer plus de tâches en même temps. Avec des instances plus petites, l’augmentation horizontale est plus fine, ce qui permet de réduire les coûts. Il n'est pas nécessaire de choisir un seul type d'instance Amazon EC2 adapté à toutes les applications de votre cluster. Au lieu de cela, vous pouvez créer plusieurs groupes Auto Scaling dans lesquels chaque groupe comporte un type d’instance différent. Vous pouvez ensuite créer un fournisseur de capacité Amazon EC2 pour chacun de ces groupes.

Utilisez les instructions suivantes pour déterminer les types de familles d’instances et le type d’instance à utiliser :

- Éliminez les types d’instances ou les familles d’instances qui ne répondent pas aux exigences spécifiques de votre application. Par exemple, si votre application nécessite un GPU, vous pouvez exclure tous les types d'instances qui n'en ont pas.
- Tenez compte des exigences, notamment en termes de débit réseau et de stockage.
- Tenez compte de l’UC et de la mémoire. En règle générale, le processeur et la mémoire doivent être suffisamment volumineux pour contenir au moins une réplique de la tâche que vous souhaitez exécuter.

Instances Spot

La capacité Spot peut permettre de réaliser d'importantes économies par rapport aux instances à la demande. La capacité Spot est une capacité excédentaire dont le prix est nettement inférieur à celui de la capacité réservée ou à la demande. Elle convient aux charges de travail de traitement par lots et de machine learning, ainsi qu'aux environnements de développement et intermédiaire. Plus généralement, elle convient à toutes les charges de travail qui tolèrent des temps d'arrêt temporaires.

Ayez conscience des conséquences suivantes, car la capacité Spot peut ne pas être disponible en permanence.

- Pendant les périodes de très forte demande, la capacité Spot peut être indisponible. Cela peut retarder le lancement des instances Spot d'Amazon EC2. Dans ce cas, les services Amazon ECS réessayent de lancer des tâches, et les groupes Amazon EC2 Auto Scaling réessayent également de lancer des instances, jusqu'à ce que la capacité requise soit disponible. Amazon EC2 ne remplace pas la capacité Spot par une capacité à la demande.
- Lorsque la demande globale de capacité augmente, les instances Spot et les tâches peuvent être résiliées avec un avertissement de deux minutes seulement. Une fois l'avertissement envoyé, les tâches doivent commencer à s'arrêter de manière ordonnée si nécessaire avant que l'instance ne soit complètement résiliée. Cela permet de réduire les risques d'erreurs. Pour plus d'informations sur un arrêt progressif, veuillez consulter le billet de blog Graceful shutdowns with ECS.

Pour réduire les pénuries de capacité Spot, tenez compte des recommandations suivantes :

- Utilisez plusieurs régions et zones de disponibilité : la capacité Spot varie en fonction de la région et de la zone de disponibilité. Vous pouvez améliorer la disponibilité Spot en exécutant vos charges de travail dans plusieurs régions et zones de disponibilité. Si possible, spécifiez des sous-réseaux dans toutes les zones de disponibilité des régions dans lesquelles vous exécutez vos tâches et instances.
- Utiliser plusieurs types d'instances Amazon EC2 : lorsque vous utilisez des stratégies d'instance mixtes avec Amazon EC2 Auto Scaling, plusieurs types d'instances sont lancés dans votre groupe Auto Scaling. Cela garantit qu'une demande de capacité Spot peut être satisfaite en cas de besoin. Pour optimiser la fiabilité et réduire la complexité, utilisez des types d'instances dotés à peu près de la même quantité de processeur et de mémoire dans votre stratégie d'instances mixtes. Ces instances peuvent provenir d'une génération différente ou de variantes du même type d'instance de base. Veuillez noter qu'elles peuvent être dotées de fonctionnalités supplémentaires dont vous n'aurez peut-être pas besoin. Une telle liste pourrait inclure, par exemple m4.large, m5.large, m5a.large, m5d.large, m5n.large, m5dn.large et m5ad.large. Pour plus d’informations, consultez Groupes Auto Scaling avec types d’instance et options d’achat multiples dans le Guide de l’utilisateur Amazon EC2 Auto Scaling.
- Utilisez la stratégie d'allocation Spot optimisée en termes de capacité : avec Amazon EC2 Spot, vous pouvez choisir entre des stratégies d'allocation optimisées en termes de capacité et de coûts. Si vous optez pour la stratégie optimisée en termes de capacités lors du lancement d'une nouvelle instance, Amazon EC2 Spot sélectionne le type d'instance le plus disponible dans la zone de disponibilité sélectionnée. Cela permet de réduire le risque que l'instance soit résiliée peu après son lancement.

Pour en savoir plus sur la configuration des avis de résiliation Spot sur vos instances de conteneur, consultez :

- Configuration des instances de conteneur Amazon ECS Linux pour recevoir des notifications relatives aux instances Spot
- Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Amazon ECS-optimized AMI Linux

> **Important**
>
> L'Amazonie ECS-Optimized L'AMI Amazon Linux 2 arrive en fin de vie le 30 juin 2026, à la même date de fin de vie que celle du système d'exploitation Amazon Linux 2 en amont (pour plus d'informations, consultez le FAQ sur Amazon Linux 2). Nous encourageons les clients à mettre à niveau leurs applications pour utiliser Amazon Linux 2023, qui inclut un support à long terme jusqu'en 2028. Pour plus d'informations sur la migration d'Amazon Linux 2 vers Amazon Linux 2023, consultez Migration depuis Amazon Linux 2 Amazon ECS-optimized AMI pour Amazon Amazon Linux 2023 ECS-optimized AMI.

Par défaut, la date de dépréciation de tous les produits Amazon ECS-optimized Les AMI sont définies deux ans après la date de création de l'AMI. Vous pouvez utiliser Amazon EC2 `DescribeImages` API pour vérifier l'état de dépréciation et la date d'une AMI. Pour plus d'informations, voir DescribeImages dans le Référence de l'API Amazon Elastic Compute Cloud.

Amazon ECS fournit l'Amazon ECS-optimized Des AMI préconfigurées avec les exigences et les recommandations nécessaires à l'exécution de vos charges de travail de conteneurs. Nous vous recommandons d'utiliser Amazon ECS-optimized AMI Amazon Linux 2023 pour vos instances Amazon EC2. Lancer vos instances de conteneurs depuis la dernière version d'Amazon ECS-Optimized AMI veille à ce que vous receviez les dernières mises à jour de sécurité et la version de l'agent de conteneur. Pour plus d'informations sur le lancement d'une instance, veuillez consulter Lancement d'une instance de conteneur Amazon ECS Linux.

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Quand vous utilisez CloudFormation pour créer un cluster, le paramètre SSM fait partie du modèle de lancement Amazon EC2 pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée pour Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, une vérification est effectuée pour voir s'il existe une mise à jour disponible qui doit être appliquée aux instances EC2. Pour un exemple d'utilisation du paramètre Systems Manager, voir Créez un cluster Amazon ECS avec Amazon ECS-optimized AMI Amazon Linux 2023 dans le AWS CloudFormation Guide de l'utilisateur.

Si vous avez besoin de personnaliser l'Amazon ECS-optimized AMI, voir Recettes de création d'AMI optimisées pour Amazon ECS sur GitHub.

Les variantes suivantes de l'Amazon ECS-optimized Les AMI sont disponibles pour vos instances Amazon EC2 dotées du système d'exploitation Amazon Linux 2023.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023	Amazon ECS-optimized AMI Amazon Linux 2023	Amazon Linux 2023 est la prochaine génération d'Amazon Linux de AWS. Recommandée dans la plupart des cas pour le lancement de vos instances Amazon EC2 pour vos charges de travail Amazon ECS. Pour plus d'informations, veuillez consulter Qu'est-ce qu'Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 (langue française non garantie).	Par défaut, Amazon ECS-optimized L'AMI Amazon Linux 2023 est fournie avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
			Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2023 est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023 (arm64)	Amazon ECS-optimized AMI Amazon Linux 2023 (arm64)	Sur la base d'Amazon Linux 2023, nous recommandons cette AMI pour lancer vos instances Amazon EC2, qui sont alimentées par Arm-based AWS Graviton/Graviton 2/Graviton 3/Graviton 4 processeurs, pour vos charges de travail Amazon ECS. Pour plus d’informations, consultez la section Spécifications relatives aux instances à usage général Amazon EC2 dans le Guide des types d’instances Amazon EC2.	Par défaut, Amazon ECS-optimized L'AMI Amazon Linux 2023 est fournie avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2023 est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d’exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2023 (Neuron)	Amazon ECS-optimized AMI Amazon Linux 2023	Basée sur Amazon Linux 2023, cette AMI est destinée aux instances Amazon EC2 Inf1, Trn1 ou Inf2. Il est livré préconfiguré avec AWS Inferentia et AWS Les conducteurs de Trainium et les AWS Neuron Runtime pour Docker qui facilite l'exécution de charges de travail d'inférence d'apprentissage automatique sur Amazon ECS. Pour plus d’informations, consultez Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones. L'Amazonie ECS-optimized L'AMI Amazon Linux 2023 (Neuron) n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon ECS-optimized L'AMI Amazon Linux 2023 est fournie avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2023 est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d’exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
GPU Amazon Linux 2023	AMI GPU Amazon Linux 2023 optimisée pour Amazon ECS	Sur la base d'Amazon Linux 2023, nous recommandons cette AMI pour lancer votre Amazon EC2 GPU-based instances pour vos charges de travail Amazon ECS. Elle est livrée préconfigurée avec les pilotes du noyau NVIDIA et une exécution de processeur graphique Docker qui font s'exécuter les applications qui tirent parti des GPU sur Amazon ECS. Pour plus d’informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.	Par défaut, Amazon ECS-optimized L'AMI Amazon Linux 2023 est fournie avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2023 est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans

Système d'exploitation	AMI	Description	Configuration du stockage
			la documentation Docker.

Les variantes suivantes de l'Amazon ECS-optimized Des AMI sont disponibles pour vos instances Amazon EC2 dotées du système d'exploitation Amazon Linux 2.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2	Amazon ECS-optimized AMI 5.10 du noyau Amazon Linux 2	Basée sur Amazon Linux 2, cette AMI est à utiliser lorsque vous lancez vos instances Amazon EC2 et que vous souhaitez utiliser le noyau Linux 5.10 au lieu du noyau 4.14 pour vos charges de travail Amazon ECS. L'Amazonie ECS-optimized L'AMI 5.10 du noyau Amazon Linux 2 n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système

Système d’exploitation	AMI	Description	Configuration du stockage
			d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2 est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2	Amazon ECS-optimized AMI Amazon Linux 2	Destinée à vos charges de travail Amazon ECS. L'Amazonie ECS-optimized L'AMI Amazon Linux 2 n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (arm64)	Amazon ECS-optimized AMI du noyau Amazon Linux 2 5.10 (arm64)	Basée sur Amazon Linux 2, cette AMI est destinée à vos instances Amazon EC2, qui sont alimentées par Arm-based AWS Graviton/Graviton 2/Graviton 3/Graviton 4 processeurs, et vous souhaitez utiliser le noyau Linux 5.10 au lieu du noyau Linux 4.14 pour vos charges de travail Amazon ECS. Pour plus d'informations, consultez la section Spécifications relatives aux instances à usage général Amazon EC2 dans le Guide des types d'instances Amazon EC2. L'Amazonie ECS-optimized L'AMI Amazon Linux 2 (arm64) n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d’exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (arm64)	Amazon ECS-optimized AMI Amazon Linux 2 (arm64)	Basée sur Amazon Linux 2, cette AMI est destinée à être utilisée lors du lancement de vos instances Amazon EC2, qui sont alimentées par Arm-based AWS Graviton/Graviton 2/Graviton 3/Graviton 4 processeurs, pour vos charges de travail Amazon ECS. L'Amazonie ECS-optimized L'AMI Amazon Linux 2 (arm64) n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d’exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (GPU)	Amazon ECS GPU-optimized noyau 5.10 AMI	Basée sur Amazon Linux 2, nous recommandons cette AMI pour lancer votre Amazon EC2 GPU-based instances avec le noyau Linux 5.10 pour vos charges de travail Amazon ECS. Elle est livrée préconfigurée avec les pilotes du noyau NVIDIA et une exécution de processeur graphique Docker qui font s'exécuter les applications qui tirent parti des GPU sur Amazon ECS. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (GPU)	Amazon ECS GPU-optimized AMI	Basée sur Amazon Linux 2, nous recommandons cette AMI pour lancer votre Amazon EC2 GPU-based instances avec le noyau Linux 4.14 pour vos charges de travail Amazon ECS. Elle est livrée préconfigurée avec les pilotes du noyau NVIDIA et une exécution de processeur graphique Docker qui font s'exécuter les applications qui tirent parti des GPU sur Amazon ECS. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon) ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (Neuron)	AMI Amazon Linux 2 (Neuron) avec noyau 5.10 optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée aux instances Inf1, Trn1 ou Inf2 d'Amazon EC2. Il est livré préconfiguré avec AWS Inferentia avec le noyau Linux 5.10 et AWS Les conducteurs de Trainium et les AWS Neuron Runtime pour Docker qui facilite l'exécution de charges de travail d'inférence d'apprentissage automatique sur Amazon ECS. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d’exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Système d'exploitation	AMI	Description	Configuration du stockage
Amazon Linux 2 (Neuron)	AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS	Basée sur Amazon Linux 2, cette AMI est destinée aux instances Inf1, Trn1 ou Inf2 d'Amazon EC2. Il est livré préconfiguré avec AWS Inferentia et AWS Les conducteurs de Trainium et les AWS Neuron Runtime pour Docker qui facilite l'exécution de charges de travail d'inférence d'apprentissage automatique sur Amazon ECS. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour AWS Charges de travail d'apprentissage automatique des neurones. L'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS n'est pas fournie avec AWS CLI préinstallé.	Par défaut, Amazon basé sur Amazon Linux 2 ECS-optimized AMI (Amazon ECS-optimized AMI Amazon Linux 2, Amazon ECS-optimized AMI Amazon Linux 2 (arm64) et Amazon ECS GPU-optimized AMI) sont livrés avec un seul volume racine de 30 Go. Vous pouvez modifier la taille du volume racine de 30 Gio lors du lancement pour augmenter le stockage disponible sur votre instance de conteneur. Ce stockage est utilisé pour le système d'exploitation, ainsi que pour les métadonnées et images Docker. Le système de fichiers par défaut pour Amazon ECS-optimized L'AMI Amazon Linux 2

Système d'exploitation	AMI	Description	Configuration du stockage
			est `xfs`, et Docker utilise le `overlay2` pilote de stockage. Pour plus d'informations, consultez Use the OverlayFS storage driver dans la documentation Docker.

Amazon ECS fournit un journal des modifications pour la variante Linux d'Amazon ECS-optimized AMI activé GitHub. Pour plus d'informations, consultez Journal des modifications.

Les variantes Linux d'Amazon ECS-optimized Les AMI utilisent l'AMI Amazon Linux 2 ou l'AMI Amazon Linux 2023 comme base. Vous pouvez récupérer le nom de l'AMI pour chaque variante en interrogeant l'API Systems Manager Parameter Store. Pour plus d'informations, consultez Récupération d'Amazon ECS-optimized Métadonnées AMI Linux. Les notes de mise à jour de l'AMI Amazon Linux 2 sont également disponibles. Pour plus d'informations, consultez Notes de mise à jour Amazon Linux 2. Les notes de mise à jour d'Amazon Linux 2023 sont également disponibles. Pour plus d'informations, veuillez consulter Notes de mise à jour Amazon Linux 2023 (langue française non garantie).

Les pages suivantes fournissent des informations supplémentaires sur les modifications :

- Source : communiqué AMI notes sur GitHub
- Notes de mise à jour Docker Engine dans la documentation Docker
- Documentation du pilote NVIDIA dans la documentation NVIDIA
- Journal des modifications de l'agent Amazon ECS sur GitHub

  Le code source de l'application `ecs-init`, ainsi que les scripts et la configuration nécessaires à l'empaquetage de l'agent, font désormais partie du référentiel de l'agent. Pour les anciennes versions de `ecs-init` et emballage, voir Journal des modifications d'Amazon ECS-Init sur GitHub

Appliquer des mises à jour de sécurité à Amazon ECS-optimized AMI

L'Amazonie ECS-optimized Les AMI basées sur Amazon Linux contiennent une version personnalisée de cloud-init. Cloud-init est un package utilisé pour démarrer des images Linux dans un environnement de cloud computing et effectuer les actions souhaitées lors du lancement d'une instance. Par défaut, tous les produits Amazon ECS-optimized Les AMI basées sur Amazon Linux publiées avant le 12 juin 2024 disposent de toutes les mises à jour de sécurité « critiques » et « importantes » appliquées au lancement de l'instance.

À compter des sorties du 12 juin 2024 d'Amazon ECS-optimized Les AMI basées sur Amazon Linux 2, le comportement par défaut n'inclura plus la mise à jour des packages au lancement. Nous vous recommandons plutôt de passer à une nouvelle version d'Amazon ECS-optimized AMI au fur et à mesure que les versions sont mises à disposition. L'Amazonie ECS-optimized Les AMI sont publiées lorsque des mises à jour de sécurité ou des modifications de base de l'AMI sont disponibles. Cela garantira que vous recevez les dernières versions des packages et mises à jour de sécurité, et que les versions des packages sont immuables lors des lancements d'instances. Pour plus d'informations sur la récupération de la dernière version d'Amazon ECS-optimized AMI, voir Récupération d'Amazon ECS-optimized Métadonnées AMI Linux.

Nous vous recommandons d'automatiser votre environnement pour qu'il soit mis à jour vers une nouvelle AMI dès qu'elle sera disponible. Pour plus d'informations sur les options disponibles, consultez le billet de blog Amazon ECS enables easier EC2 capacity management, with managed instance draining.

Pour continuer à appliquer manuellement les mises à jour de sécurité « critiques » et « importantes » sur une version d'AMI, vous pouvez exécuter la commande suivante sur votre instance Amazon EC2.

```
yum update --security
```

> **Warning**
>
> La mise à jour des packages docker ou containerd arrêtera tous les conteneurs en cours d'exécution sur l'hôte, ce qui signifie que toutes les tâches Amazon ECS en cours d'exécution seront arrêtées. Planifiez en conséquence pour minimiser les interruptions de service.

Si vous souhaitez réactiver les mises à jour de sécurité au lancement, vous pouvez ajouter la ligne suivante à la section `#cloud-config` des données utilisateur de cloud-init lors du lancement de

votre instance Amazon EC2. Pour plus d'informations, consultez la section Utilisation de cloud-init sur Amazon Linux 2 dans le Guide de l'utilisateur Amazon Linux.

```
#cloud-config
repo_upgrade: security
```

Version-locked colis sur Amazon ECS-optimized AMI GPU AL2023

Certains packages sont essentiels au bon fonctionnement et aux performances des fonctionnalités du GPU sur Amazon ECS-optimized AMI GPU AL2023. Il s'agit des licences suivantes :

- Pilotes NVIDIA (`nvidia*`)
- Modules du noyau (`kmod*`)
- Bibliothèques NVIDIA (`libnvidia*`)
- Packages du noyau (`kernel*`)
- Packages du serveur Xorg (`xorg*`)
- Gestionnaire de processeurs graphiques pour centres de données NVIDIA (DCGM) (`datacenter-gpu-manager-*`)

> **Note**
>
> Cette liste n'est pas exhaustive. La liste complète des packages verrouillés est disponible sur `dnf versionlock list`

Ces packages sont verrouillés pour garantir la stabilité et empêcher les modifications involontaires susceptibles de perturber les charges de travail du GPU. Par conséquent, ces packages doivent généralement être modifiés dans les limites d'un processus géré qui gère correctement les problèmes potentiels et maintient les fonctionnalités du GPU.

Pour éviter toute modification involontaire, `dnf versionlock` le plugin est utilisé sur ces packages.

Si vous souhaitez modifier un package verrouillé, vous pouvez :

```
# unlock a single package
sudo dnf versionlock delete $PACKAGE_NAME
```

```
# unlock all packages
sudo dnf versionlock clear
```

> **Important**
>
> Lorsque des mises à jour de ces packages sont nécessaires, les clients doivent envisager d'utiliser la dernière version d'AMI qui inclut les mises à jour requises. Si la mise à jour d'instances existantes est requise, il convient d'adopter une approche prudente impliquant le déverrouillage, la mise à jour et le reverrouillage des packages, en veillant toujours à ce que les fonctionnalités du GPU soient maintenues tout au long du processus.

Récupération d'Amazon ECS-optimized Métadonnées AMI Linux

Vous pouvez récupérer l'Amazon par programmation ECS-optimized Métadonnées AMI. Les métadonnées incluent le nom de l'AMI, la version de l'agent de conteneur Amazon ECS et la version de l'exécution Amazon ECS qui inclut la version Docker.

Lorsque vous créez un cluster à l'aide de la console, Amazon ECS crée un modèle de lancement pour vos instances avec la dernière AMI associée au système d'exploitation sélectionné.

Quand vous utilisez CloudFormation pour créer un cluster, le paramètre SSM fait partie du modèle de lancement Amazon EC2 pour les instances du groupe Auto Scaling. Vous pouvez configurer le modèle pour utiliser un paramètre dynamique de Systems Manager afin de déterminer l'AMI optimisée pour Amazon ECS à déployer. Ce paramètre garantit que chaque fois que vous déployez la pile, une vérification est effectuée pour voir s'il existe une mise à jour disponible qui doit être appliquée aux instances EC2. Pour un exemple d'utilisation du paramètre Systems Manager, voir Créez un cluster Amazon ECS avec Amazon ECS-optimized AMI Amazon Linux 2023 dans le AWS CloudFormation Guide de l'utilisateur.

L'ID AMI, le nom de l'image, le système d'exploitation, la version de l'agent de conteneur, le nom de l'image source et la version d'exécution pour chaque variante d'Amazon ECS-optimized Les AMI peuvent être récupérées par programmation en interrogeant l'API Systems Manager Parameter Store. Pour plus d'informations sur l'API Systems Manager Parameter Store, voir GetParameters et GetParametersByPath.

> **Note**
>
> Votre utilisateur administratif doit disposer des autorisations IAM suivantes pour récupérer l'Amazon ECS-optimized Métadonnées AMI. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`.
>
> - sms :GetParameters
> - sms :GetParameter
> - sms :GetParametersByPath

Format de paramètre Systems Manager Parameter Store

Voici le format du nom des paramètres pour chaque Amazon ECS-optimized Variante AMI.

Amazon pour Linux ECS-optimized AMI

- Métadonnées d'AMI Amazon Linux 2023 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/<version>
```

- Métadonnées d'AMI Amazon Linux 2023 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/arm64/<version>
```

- Métadonnées d'AMI Amazon Linux 2023 (Neuron) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/neuron/<version>
```

- Métadonnées de l’AMI Amazon Linux 2023 (GPU) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/<version>
```

  Métadonnées d'AMI Amazon Linux 2 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/<version>
```

- Métadonnées d'AMI Amazon Linux 2 noyau 5.10 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/arm64/<version>
```

- Métadonnées d'AMI Amazon Linux 2 noyau 5.10 (arm64) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/arm64/<version>
```

- Amazon ECS GPU-optimized métadonnées AMI du noyau 5.10 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/gpu/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (GPU) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/gpu/<version>
```

- Métadonnées de l’AMI Amazon Linux 2 (Neuron) avec noyau 5.10 optimisée pour Amazon ECS :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/inf/<version>
```

- Métadonnées d'AMI Amazon Linux 2 (Neuron) :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/inf/<version>
```

Le format de nom de paramètre suivant permet de récupérer l'ID d'image de la dernière version recommandée d'Amazon ECS-optimized AMI Amazon Linux 2 à l'aide du sous-paramètre `image_id`.

```
/aws/service/ecs/optimized-ami/amazon-linux-2/recommended/image_id
```

Le format de nom de paramètre suivant permet de récupérer les métadonnées d'un Amazon spécifique ECS-optimized Version de l'AMI en spécifiant le nom de l'AMI.

- Amazon ECS-optimized Métadonnées AMI Amazon Linux 2 :

```
/aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-ecs-hvm-2.0.20181112-x86_64-ebs
```

> **Note**
>
> Toutes les versions d'Amazon ECS-optimized Les AMI Amazon Linux 2 peuvent être récupérées. Amazon uniquement ECS-optimized Versions de l'AMI `amzn-ami-2017.09.l-amazon-ecs-optimized` (Linux) et versions ultérieures peuvent être récupérées.

Exemples

Les exemples suivants montrent comment vous pouvez récupérer les métadonnées de chaque Amazon. ECS-optimized Variante AMI.

Récupération des métadonnées de la dernière version recommandée d'Amazon ECS-optimized AMI

Vous pouvez récupérer la dernière version recommandée d'Amazon ECS-optimized AMI utilisant le AWS CLI avec les éléments suivants AWS CLI commandes.

Amazon pour Linux ECS-optimized AMI

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2023 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2023 (arm64) :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
arm64/recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2023 (Neuron) :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
neuron/recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI GPU Amazon Linux 2023 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/gpu/
recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI du noyau 5.10 d'Amazon Linux 2 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI du noyau 5.10 (arm64) d'Amazon Linux 2 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/arm64/recommended --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2 (arm64) :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/arm64/
recommended --region us-east-1
```

- Pour Amazon ECS GPU-optimized AMI du noyau 5.10 :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/gpu/recommended --region us-east-1
```

- Pour Amazon ECS GPU-optimized AMI :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/gpu/
recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 (Neuron) avec noyau 5.10 optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/inf/recommended --region us-east-1
```

- Pour les AMI Amazon Linux 2 (Neuron) optimisées pour Amazon ECS :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/inf/
recommended --region us-east-1
```

Récupération de l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized AMI Amazon Linux 2023

Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized ID AMI Amazon Linux 2023 à l'aide du sous-paramètre `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-
linux-2023/recommended/image_id --region us-east-1
```

Pour extraire uniquement la valeur `image_id`, vous pouvez interroger la valeur de paramètre spécifique ; par exemple :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2023/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Récupération des métadonnées d'un Amazon spécifique ECS-optimized Version AMI d'Amazon Linux 2

Récupérez les métadonnées d'un Amazon spécifique ECS-optimized Version de l'AMI Amazon Linux utilisant le AWS CLI avec les éléments suivants AWS CLI commande. Remplacez le nom de l'AMI par le nom de l'Amazon ECS-optimized AMI Amazon Linux à récupérer.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/amzn2-ami-
ecs-hvm-2.0.20200928-x86_64-ebs --region us-east-1
```

Récupération de l'Amazon ECS-optimized Métadonnées AMI du noyau 5.10 d'Amazon Linux 2 à l'aide du gestionnaire de systèmes GetParametersByPath API

Récupérez l'Amazon ECS-optimized Métadonnées AMI Amazon Linux 2 avec le gestionnaire de systèmes GetParametersByPath API utilisant le AWS CLI à l'aide de la commande suivante.

```
aws ssm get-parameters-by-path --path /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/ --region us-east-1
```

Récupération de l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized AMI 5.10 du noyau Amazon Linux 2

Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized ID d'AMI du noyau 5.10 d'Amazon Linux 2 à l'aide du sous-paramètre `image_id`.

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
kernel-5.10/recommended/image_id --region us-east-1
```

Pour extraire uniquement la valeur `image_id`, vous pouvez interroger la valeur de paramètre spécifique ; par exemple :

```
aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
recommended/image_id --region us-east-1 --query "Parameters[0].Value"
```

Utilisation de la dernière version recommandée d'Amazon ECS-optimized AMI dans un CloudFormation modèle

Vous pouvez consulter la dernière version recommandée d'Amazon ECS-optimized AMI dans un CloudFormation modèle en faisant référence au nom du magasin de paramètres de Systems Manager.

Exemple Linux

```
Parameters:kernel-5.10
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ecs/optimized-ami/amazon-linux-2/kernel-5.10/recommended/
image_id
```

Migration d'un Amazon Linux 2 vers un Amazon Amazon Linux 2023 ECS-optimized AMI

Suivant Amazon Linux, Amazon ECS met fin à la prise en charge standard d'Amazon Linux 2 Amazon ECS-optimized AMI en vigueur le 30 juin 2026. Après cette date, la version de l'agent Amazon ECS est épinglée et le nouvel Amazon Linux 2 Amazon ECS-optimized Les AMI ne sont publiées que lorsque l'AMI source Amazon Linux 2 est mise à jour. La fin de vie complète (EOL) aura lieu le 30 juin 2026, après quoi il n'y aura plus d'Amazon ECS-optimized Les AMI Amazon Linux 2 sont publiées, même si l'AMI source est mise à jour.

Amazon Linux 2023 propose une approche sécurisée par défaut avec des politiques de sécurité préconfigurées, SELinux en mode permissif, IMDSv2-only mode activé par défaut, temps de démarrage optimisés et gestion améliorée des packages pour une sécurité et des performances accrues.

Il existe un haut degré de compatibilité entre Amazon Linux 2 et Amazon Linux 2023 (Amazon). ECS-optimized Les AMI et la plupart des clients connaîtront des modifications minimes, voire nulles, de leurs charges de travail entre les deux systèmes d'exploitation.

Pour plus d'informations, consultez la section Comparaison entre Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023 et les FAQ AL2023.

Considérations de compatibilité

Gestion des packages et mises à jour du système d'exploitation

Contrairement aux versions précédentes d'Amazon Linux, Amazon ECS-optimized Les AMI Amazon Linux 2023 sont verrouillées sur une version spécifique du référentiel Amazon Linux. Cela permet aux utilisateurs de ne pas mettre à jour par inadvertance des packages susceptibles d'apporter des modifications indésirables ou irréversibles. Pour plus d'informations, consultez la section Gestion des référentiels et des mises à jour du système d'exploitation dans Amazon Linux 2023 du Guide de l'utilisateur Amazon Linux 2023.

Versions du noyau Linux

Les AMI Amazon Linux 2 sont basées sur les noyaux Linux 4.14 et 5.10, tandis qu'Amazon Linux 2023 utilise les noyaux Linux 6.1 et 6.12. Pour plus d'informations, consultez la section Comparaison des noyaux Amazon Linux 2 et Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023.

Modifications relatives à la disponibilité des packages

Ci-dessous figurent les modifications notables apportées aux paquets dans Amazon Linux 2023 :

- Certains paquets binaires source d'Amazon Linux 2 ne sont plus disponibles dans Amazon Linux 2023. Pour plus d'informations, consultez la section Packages supprimés d'Amazon Linux 2023 dans les Notes de mise à jour d'Amazon Linux 2023.
- Modifications apportées à la manière dont Amazon Linux prend en charge les différentes versions des packages. Le système `amazon-linux-extras` utilisé dans Amazon Linux 2 n'existe pas dans Amazon Linux 2023. Tous les packages sont simplement disponibles dans le référentiel « core ».
- Les packages supplémentaires pour Enterprise Linux (EPEL) ne sont pas pris en charge dans Amazon Linux 2023. Pour plus d'informations, consultez la section Compatibilité EPEL dans Amazon Linux 2023 du Guide de l'utilisateur Amazon Linux 2023.

- Les applications 32 bits ne sont pas prises en charge dans Amazon Linux 2023. Pour plus d'informations, consultez la section Fonctionnalités obsolètes d'Amazon Linux 2 dans le Guide de l'utilisateur Amazon Linux 2023.

Modifications relatives aux groupes de contrôle (cgroups)

Un groupe de contrôle (cgroup) est une fonctionnalité du noyau Linux permettant d'organiser hiérarchiquement les processus et de répartir les ressources système entre eux. Les groupes de contrôle sont très fréquemment utilisés pour implémenter un environnement d'exécution de conteneur, et par `systemd`.

Amazon Linux 2023 utilise cgroupv2 par défaut, tandis qu'Amazon Linux 2 utilise cgroupv1. L'agent Amazon ECS, Docker et containerd prennent tous en charge à la fois cgroupv1 et cgroupv2. Pour plus de détails sur cgroupv2, consultez la section Groupes de contrôle v2 dans Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023.

Rapports sur l'utilisation de la mémoire

cgroupv2 modifie la façon dont l'utilisation de la mémoire des conteneurs est calculée. Dans cgroupv1 (Amazon Linux 2), l'utilisation de la mémoire du conteneur telle que signalée par le moteur d'exécution du conteneur exclut généralement le cache de pages. Dans cgroupv2 (Amazon Linux 2023), le cache des pages est inclus dans l'utilisation de mémoire signalée. La même charge de travail peut entraîner une utilisation de la mémoire supérieure sur Amazon Linux 2023 par rapport à Amazon Linux 2, même lorsque la consommation réelle de mémoire de l'application n'a pas changé.

Nous vous recommandons d'évaluer l'utilisation de la mémoire sur les instances Amazon Linux 2023 avant de migrer les charges de travail de production, et d'ajuster les limites de mémoire des tâches et des conteneurs si nécessaire. Vous pouvez utiliser Informations sur les conteneurs pour comparer l'utilisation de la mémoire entre Amazon Linux 2 et Amazon Linux 2023.

La mémoire des tâches limite la visibilité

Lorsque vous définissez la mémoire au niveau de la tâche sans définir de limite de mémoire au niveau du conteneur, Amazon ECS applique la limite au cgroup de la tâche. Cela constitue la limite de mémoire effective pour tous les conteneurs appartenant à la tâche. Sur cgroupv1 (Amazon Linux 2), les cgroups enfants pouvaient lire la limite de mémoire effective à leur propre niveau. Ils n'avaient pas besoin de parcourir la hiérarchie des groupes de contrôle. Sur cgroupv2 (Amazon Linux 2023), la limite effective n'est pas visible pour le conteneur. De plus, Docker exécute les conteneurs dans

un espace de noms de groupe de contrôle privé par défaut, ce qui les empêche de parcourir la hiérarchie des groupes de contrôle pour découvrir la limite du parent.

Tout processus à l'intérieur du conteneur qui lit les limites de mémoire du cgroup pour déterminer la mémoire disponible verra la mémoire hôte complète plutôt que la limite au niveau des tâches. Par exemple, la machine virtuelle Java (JVM) est connue pour être affectée par ce problème. Il utilise les limites des groupes de contrôle pour dimensionner automatiquement le tas. Lorsqu'il ne peut pas détecter la véritable limite, il suralloue la mémoire, ce qui augmente le risque de destruction par manque de mémoire (OOM). D'autres environnements d'exécution ou frameworks qui lisent les limites des groupes de contrôle pour les décisions relatives aux ressources peuvent être affectés de la même manière.

Cela se produit lorsque la mémoire au niveau des tâches est définie mais au niveau du conteneur `memory` n'est pas défini dans la définition de la tâche.

Pour résoudre ce problème, choisissez l'une des solutions suivantes :

Définissez des limites de mémoire au niveau du conteneur (recommandé)

Définissez explicitement le `memory` paramètre au niveau du conteneur dans votre définition de tâche, égal à la valeur de mémoire au niveau de la tâche. Cette approche est sûre, fiable et ne nécessite aucune modification de l'infrastructure.

Activer la propagation des limites de mémoire des tâches au niveau de l'agent

Réglez le `ECS_PROPAGATE_TASK_MEMORY_LIMIT_CGROUPV2` variable d'environnement à `true` sur l'agent Amazon ECS. Lorsqu'il est activé et sur cgroupv2, l'agent définit automatiquement la limite de mémoire de chaque conteneur à la valeur au niveau de la tâche si le conteneur ne dispose pas déjà d'une limite de mémoire explicite au niveau du conteneur. La limite de mémoire des tâches est ainsi visible à l'intérieur du conteneur sans qu'il soit nécessaire de modifier la définition des tâches.

Vous pouvez définir cela dans le fichier de configuration de l'agent Amazon ECS (`/etc/ecs/ecs.config`) sur vos instances Amazon EC2 :

```
ECS_PROPAGATE_TASK_MEMORY_LIMIT_CGROUPV2=true
```

Cette option nécessite la version 1.104.0 ou ultérieure de l'agent Amazon ECS. Cela ne s'applique qu'aux instances Linux utilisant cgroupv2. Il n'est pas pris en charge sous Windows.

Modifications relatives au service de métadonnées d'instance (IMDS)

Amazon Linux 2023 nécessite la version 2 du service de métadonnées d'instance (IMDSv2) par défaut. L'IMDSv2 présente plusieurs avantages qui contribuent à améliorer la posture de sécurité. Il utilise une méthode d'authentification orientée session qui nécessite la création d'un jeton secret dans une simple requête HTTP PUT pour démarrer la session. La durée de validité du jeton de session peut avoir une valeur quelconque entre 1 seconde et 6 heures.

Pour plus d'informations sur la transition de IMDSv1 à IMDSv2, consultez la section Transition vers l'utilisation du service de métadonnées d'instance version 2 dans le Guide de l'utilisateur Amazon EC2.

Si vous souhaitez utiliser IMDSv1, vous pouvez toujours le faire en remplaçant manuellement les paramètres à l'aide des propriétés de lancement des options de métadonnées de l'instance.

Modifications relatives à la permutation de mémoire

Per-container l'échange de mémoire n'est pas pris en charge sur Amazon Linux 2023 et cgroups v2. Pour plus d'informations, consultez Gestion de l'espace mémoire d'échange de conteneur sur Amazon ECS.

Modifications relatives à la validation FIPS

Amazon Linux 2 est certifié selon la norme FIPS 140-2 et Amazon Linux 2023 selon la norme FIPS 140-3.

Pour activer le mode FIPS sur Amazon Linux 2023, installez les packages nécessaires sur votre instance Amazon EC2 et suivez les étapes de configuration en suivant les instructions de la section Activer le mode FIPS sur Amazon Linux 2023 dans le Guide de l'utilisateur Amazon Linux 2023.

Prise en charge des instances accélérées

L'Amazonie ECS-optimized Les AMI Amazon Linux 2023 prennent en charge à la fois les types d'instances accélérées par Neuron et par GPU. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

Création d'AMI personnalisées

Bien que nous vous recommandons de passer à Amazon officiellement pris en charge et publié ECS-optimized AMI pour Amazon Linux 2023, vous pouvez continuer à créer Amazon Linux 2 Amazon

personnalisé ECS-optimized AMI utilisant les scripts de construction open source utilisés pour créer les variantes Linux d'Amazon ECS-optimized AMI. Pour plus d’informations, consultez Amazon ECS-optimized Script de génération d'AMI pour Linux.

Stratégies de migration

Nous vous recommandons de créer et de mettre en œuvre un plan de migration incluant des tests approfondis des applications. Les sections suivantes décrivent les différentes stratégies de migration en fonction de la façon dont vous gérez votre infrastructure Amazon ECS.

Migration avec des fournisseurs de capacité Amazon ECS

1. Créez un fournisseur de capacité à l’aide d’un modèle de lancement. Cela devrait faire référence à un groupe Auto Scaling avec un modèle de lancement similaire à votre groupe existant, mais au lieu d'Amazon Linux 2 Amazon ECS-optimized AMI, il doit spécifier l'une des variantes d'Amazon Linux 2023. Ajoutez ce nouveau fournisseur de capacité à votre cluster Amazon ECS existant.
2. Mettez à jour la stratégie de fournisseur de capacité par défaut de votre cluster pour inclure à la fois le fournisseur de capacité Amazon Linux 2 existant et le nouveau fournisseur de capacité Amazon Linux 2023. Commencez par un poids plus élevé chez le fournisseur Amazon Linux 2 et un poids inférieur chez le fournisseur Amazon Linux 2023 (par exemple, Amazon Linux 2 : poids de 80, Amazon Linux 2023 : poids de 20). Amazon ECS commence alors à allouer les instances Amazon Linux 2023 au fur et à mesure que de nouvelles tâches sont planifiées. Vérifiez que les instances s’enregistrent correctement et que les tâches peuvent s’exécuter correctement sur les nouvelles instances.
3. Ajustez progressivement les pondérations des fournisseurs de capacité dans la stratégie par défaut de votre cluster, en augmentant la pondération du fournisseur Amazon Linux 2023 tout en diminuant la pondération du fournisseur Amazon Linux 2 au fil du temps (par exemple, 60/40, puis 40/60, puis 20/80). Vous pouvez également mettre à jour les stratégies de chaque fournisseur de capacité de service afin de prioriser les instances Amazon Linux 2023. Surveillez le placement des tâches pour vous assurer qu’elles s’exécutent correctement sur les instances Amazon Linux 2023.
4. Drainez éventuellement les instances de conteneur Amazon Linux 2 pour accélérer la migration des tâches. Si vous disposez d'une capacité de remplacement suffisante pour Amazon Linux 2023, vous pouvez vider manuellement vos instances de conteneur Amazon Linux 2 via la console Amazon ECS ou AWS CLI pour accélérer la transition de vos tâches d'Amazon Linux 2 vers Amazon Linux 2023. Une fois la migration terminée, supprimez le fournisseur de capacité Amazon Linux 2 de votre cluster et supprimez le groupe Auto Scaling associé.

Migration à l’aide d’un groupe Amazon EC2 Auto Scaling

1. Créez un nouveau groupe Amazon EC2 Auto Scaling avec un nouveau modèle de lancement. Il doit être similaire à votre modèle de lancement existant, mais à la place d'Amazon Linux 2, Amazon ECS-optimized AMI, il doit spécifier l'une des variantes d'Amazon Linux 2023. Ce nouveau groupe Auto Scaling peut lancer des instances sur votre cluster existant.
2. Augmentez verticalement le groupe Auto Scaling afin que les instances Amazon Linux 2023 commencent à s’enregistrer dans votre cluster. Vérifiez que les instances s’enregistrent correctement et que les tâches peuvent s’exécuter correctement sur les nouvelles instances.
3. Une fois que vos tâches ont été vérifiées pour fonctionner sur Amazon Linux 2023, augmentez verticalement le groupe Auto Scaling Amazon Linux 2023 tout en réduisant verticalement de manière progressive le groupe Auto Scaling Amazon Linux 2, jusqu’à ce que vous ayez complètement remplacé toutes les instances Amazon Linux 2.
4. Si vous disposez d’une capacité de remplacement suffisante pour Amazon Linux 2023, vous souhaiterez peut-être drainer explicitement les instances de conteneur afin d’accélérer la transition de vos tâches d’Amazon Linux 2 vers Amazon Linux 2023. Pour plus d’informations, consultez Drainage des instances de conteneur Amazon ECS.

Migration à l’aide d’instances gérées manuellement

1. Lancez manuellement (ou ajustez les scripts qui lancent) de nouvelles instances Amazon EC2 à l'aide d'Amazon ECS-optimized AMI Amazon Linux 2023 au lieu d'Amazon Linux 2. Assurez-vous que ces instances utilisent les mêmes groupes de sécurité, sous-réseaux, rôles IAM et configuration de cluster que vos instances Amazon Linux 2 existantes. Les instances devraient s’enregistrer automatiquement dans votre cluster Amazon ECS existant lors du lancement.
2. Vérifiez que les nouvelles instances Amazon Linux 2023 sont correctement enregistrées dans votre cluster Amazon ECS et qu’elles sont dans l’état `ACTIVE`. Vérifiez que les tâches peuvent être planifiées et exécutées correctement sur ces nouvelles instances soit en attendant le placement naturel des tâches, soit manuellement stopping/starting certaines tâches pour déclencher une replanification.
3. Remplacez progressivement vos instances Amazon Linux 2 en lançant des instances Amazon Linux 2023 supplémentaires selon les besoins, puis en drainant et en arrêtant manuellement les instances Amazon Linux 2 une par une. Vous pouvez drainer les instances via la console Amazon ECS en définissant le statut de l’instance sur `DRAINING`, ce qui empêchera d’y placer de nouvelles tâches et permettra aux tâches existantes de se terminer ou d’être reprogrammées ailleurs.

Amazon ECS-optimized Script de génération d'AMI pour Linux

Amazon ECS a ouvert les scripts de génération utilisés pour créer les variantes Linux d'Amazon ECS-optimized AMI. Ces scripts de construction sont désormais disponibles sur GitHub. Pour plus d'informations, voir amazon ecs-ami sur GitHub.

Si vous avez besoin de personnaliser l'Amazon ECS-optimized AMI, voir Recettes de création d'AMI optimisées pour Amazon ECS sur GitHub.

Le référentiel de scripts de construction comprend un HashiCorp emballeur créer et créer des scripts pour générer chacune des variantes Linux d'Amazon ECS-optimized AMI. Ces scripts sont la source de vérité pour Amazon ECS-optimized AMI se construit, afin que vous puissiez suivre les GitHub référentiel pour suivre les modifications apportées à nos AMI. Par exemple, vous pouvez souhaiter que votre propre AMI utilise la même version de Docker que celle utilisée par l'équipe Amazon ECS pour l'AMI officielle.

Pour plus d'informations, consultez le référentiel Amazon ECS AMI à l'adresse aws/amazon-ecs-ami sur GitHub.

Pour créer une Amazon ECS-optimized AMI Linux

1. Clonez le `aws/amazon-ecs-ami` GitHub repo.

   ```
   git clone https://github.com/aws/amazon-ecs-ami.git
   ```

2. Ajoutez une variable d'environnement pour AWS Région à utiliser lors de la création de l'AMI. Remplacez la valeur `us-west-2` avec la Région à utiliser.

   ```
   export REGION=us-west-2
   ```

3. Un Makefile est fourni pour créer l'AMI. Depuis le répertoire racine du dépôt cloné, utilisez l'une des commandes suivantes, correspondant à la variante Linux d'Amazon ECS-optimized AMI que vous souhaitez créer.

   - Amazon ECS-optimized AMI Amazon Linux 2

     ```
     make al2
     ```

   - Amazon ECS-optimized AMI Amazon Linux 2 (arm64)

```
make al2arm
```

- Amazon ECS GPU-optimized AMI

```
make al2gpu
```

- AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS

```
make al2inf
```

- Amazon ECS-optimized AMI Amazon Linux 2023

```
make al2023
```

- Amazon ECS-optimized AMI Amazon Linux 2023 (arm64)

```
make al2023arm
```

- Amazon ECS-optimized AMI GPU Amazon Linux 2023

```
make al2023gpu
```

- AMI Amazon Linux 2023 (Neuron) optimisée pour Amazon ECS

```
make al2023neu
```

Amazon ECS-optimized Fusée à bouteilles AMI

Bottlerocket est un Linux système d'exploitation open source basé sur des bases spécialement conçues par AWS pour exécuter des conteneurs sur des machines virtuelles ou des hôtes bare metal. L'Amazonie ECS-optimized Bottlerocket AMI est sécurisé et ne comprend que le nombre minimum de packages requis pour faire fonctionner des conteneurs. Cela améliore l'utilisation des ressources, réduit la surface d'attaque de sécurité et contribue à réduire les frais de gestion. L'AMI Bottlerocket est également intégrée à Amazon ECS afin de réduire la charge opérationnelle liée à la mise à jour des instances de conteneur dans un cluster.

Bottlerocket diffère d'Amazon Linux par les aspects suivants :

- Bottlerocket n'inclut pas de gestionnaire de packages et son logiciel ne peut être exécuté que sous forme de conteneurs. Les mises à jour de Bottlerocket sont appliquées et peuvent être annulées en une seule étape, ce qui réduit le risque d'erreurs de mise à jour.
- Le principal mécanisme de gestion des hôtes Bottlerocket est un planificateur de conteneurs. Contrairement à Amazon Linux, la connexion à des instances Bottlerocket individuelles est censée être une opération peu fréquente uniquement à des fins avancées de débogage et de dépannage.

Pour plus d'informations sur Bottlerocket, consultez la documentation et les versions sur GitHub.

Il existe des variantes de l'Amazon ECS-optimized Bottlerocket AMI pour le noyau 6.1 et le noyau 5.10.

Les variantes suivantes utilisent le noyau 6.1 :

- `aws-ecs-2`
- `aws-ecs-2-nvidia`

Les variantes suivantes utilisent le noyau 5.10 :

- `aws-ecs-1`
- `aws-ecs-1-nvidia`

  Pour plus d'informations sur la variante `aws-ecs-1-nvidia`, veuillez consulter le billet de blog Announcing NVIDIA GPU support for Bottlerocket on Amazon ECS.

Considérations

Tenez compte des informations suivantes lors de l'utilisation d'une AMI Bottlerocket avec Amazon ECS.

- Bottlerocket prend en charge les instances Amazon EC2 avec processeurs `x86_64` et `arm64`. Il n'est pas recommandé d'utiliser une AMI Bottlerocket avec les instances Amazon EC2 dotées d'une puce Inferentia.
- Les images Bottlerocket n'incluent pas un serveur SSH ou un shell. Vous pouvez toutefois utiliser des outils de gestion hors bande pour obtenir un accès administrateur SSH et effectuer un amorçage.

Pour plus d'informations, consultez ces sections dans le fusée à bouteilles README.md sur GitHub:

- Exploration
- Conteneur d'administration

- Par défaut, Bottlerocket possède un conteneur de contrôle qui est activé. Ce conteneur exécute l'agent AWS Systems Manager que vous pouvez utiliser pour exécuter des commandes ou démarrer des sessions shell sur les instances Bottlerocket d'Amazon EC2. Pour plus d'informations, consultez Configuration du gestionnaire de session dans le Guide de l'utilisateur AWS Systems Manager .

- Bottlerocket est optimisé pour les applications de conteneurs et met l'accent sur la sécurité. Bottlerocket n'inclut pas de gestionnaire de package et est inaltérable.

  Pour plus d'informations sur les fonctionnalités de sécurité et les conseils, voir Caractéristiques de sécurité et Directives de sécurité sur GitHub.
- Le mode réseau `awsvpc` est pris en charge pour la version d'AMI Bottlerocket `1.1.0` ou version ultérieure.
- App Mesh dans une définition de tâche est pris en charge pour la version d'AMI Bottlerocket `1.15.0` ou ultérieure.
- Le paramètre de définition de tâche `initProcessEnabled` est pris en charge pour la version `1.19.0` ou ultérieure de l'AMI Bottlerocket.
- Les AMI Bottlerocket ne prennent pas non plus en charge les services et fonctionnalités suivants :
  - ECS Anywhere
  - Connexion au service
  - Amazon EFS en mode chiffré
  - Amazon EFS en mode réseau `awsvpc`
  - Les volumes Amazon EBS ne peuvent pas être montés
  - Accélérateur Elastic Inference

Récupération d'Amazon ECS-optimized Fusée à bouteilles Métadonnées AMI

Vous pouvez récupérer l'identifiant Amazon Machine Image (AMI) pour Amazon ECS-optimized AMI en interrogeant le AWS Systems Manager API de stockage des paramètres. Avec ce paramètre,

vous n'avez pas besoin de rechercher manuellement Amazon ECS-optimized Identifiants AMI. Pour plus d'informations sur l'API Systems Manager Parameter Store, voir GetParameter. L'utilisateur que vous utilisez doit disposer du `ssm:GetParameter` Autorisation IAM pour récupérer l'Amazon ECS-optimized Métadonnées AMI.

`aws-ecs-2` Variante Bottlerocket AMI

Vous pouvez récupérer la dernière version stable `aws-ecs-2` Variante Bottlerocket AMI de Région AWS et architecture avec le AWS CLI ou le Console de gestion AWS.

- AWS CLI — Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized Bottlerocket AMI avec les éléments suivants AWS CLI commande à l'aide du sous-paramètre `image_id`. Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-2 --name "/aws/service/bottlerocket/aws-ecs-2/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- Console de gestion AWS — Vous pouvez rechercher l'Amazon recommandé ECS-optimized ID AMI à l'aide d'une URL dans Console de gestion AWS. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/aws-ecs-2/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
aws-ecs-2/arm64/latest/image_id/description?region=region#
```

aws-ecs-2-nvidia Variante Bottlerocket AMI

Vous pouvez récupérer la dernière version stable `aws-ecs-2-nvdia` Variante Bottlerocket AMI par région et architecture avec AWS CLI ou le Console de gestion AWS.

- AWS CLI — Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized Bottlerocket AMI avec les éléments suivants AWS CLI commande à l'aide du sous-paramètre `image_id`. Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-2-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- Console de gestion AWS : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' Console de gestion AWS. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-2-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

```
https://regionconsole.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-2-nvidia/arm64/latest/image_id/description?region=region#
```

aws-ecs-1 Variante Bottlerocket AMI

Vous pouvez récupérer la dernière version stable `aws-ecs-1` Variante Bottlerocket AMI de Région AWS et architecture avec le AWS CLI ou le Console de gestion AWS.

- AWS CLI — Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized Bottlerocket AMI avec les éléments suivants AWS CLI commande à l'aide du sous-paramètre `image_id`. Remplacez la *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub. Pour récupérer une autre version que la dernière, remplacez `latest` par le numéro de version.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- Console de gestion AWS — Vous pouvez rechercher l'Amazon recommandé ECS-optimized ID AMI à l'aide d'une URL dans Console de gestion AWS. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *`region`* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
    bottlerocket/aws-ecs-1/x86_64/latest/image_id/description
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

```
https://region.console.aws.amazon.com/systems-manager/parameters/aws/service/
bottlerocket/aws-ecs-1/arm64/latest/image_id/description
```

aws-ecs-1-Nvidia Variante Bottlerocket AMI

Vous pouvez récupérer la dernière version stable `aws-ecs-1-nvdia` Variante Bottlerocket AMI par région et architecture avec AWS CLI ou le Console de gestion AWS.

- AWS CLI — Vous pouvez récupérer l'identifiant de l'image de la dernière version recommandée d'Amazon ECS-optimized Bottlerocket AMI avec les éléments suivants AWS CLI commande à l'aide du sous-paramètre `image_id`. Remplacez la *region* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/x86_64/latest/image_id" --query Parameter.Value --output text
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

    ```
    aws ssm get-parameter --region us-east-1 --name "/aws/service/bottlerocket/aws-
    ecs-1-nvidia/arm64/latest/image_id" --query Parameter.Value --output text
    ```

- Console de gestion AWS : vous pouvez interroger l'ID d'AMI optimisée pour Amazon ECS recommandée à l'aide d'une URL dans l' Console de gestion AWS. L'URL ouvre la console Amazon EC2 Systems Manager avec la valeur de l'ID du paramètre. Dans l'URL suivante, remplacez *region* par le code de région pour lequel vous souhaitez obtenir l'ID AMI.

  Pour plus d'informations sur les Régions AWS, voir Trouver une AMI sur GitHub.

  - Pour l'architecture 64 bits (`x86_64`) :

    ```
    https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
    aws-ecs-1-nvidia/x86_64/latest/image_id/description?region=region#
    ```

  - Pour l'architecture Arm 64 bits (`arm64`) :

```
https://console.aws.amazon.com/systems-manager/parameters/aws/service/bottlerocket/
aws-ecs-1-nvidia/arm64/latest/image_id/description?region=region#
```

Étapes suivantes

Pour un didacticiel détaillé sur la façon de démarrer avec Bottlerocket système d'exploitation sur Amazon ECS, voir Utilisation d'une AMI Bottlerocket avec Amazon ECS sur GitHub et Commencer avecBottlerocket et Amazon ECS sur le AWS site de blog.

Pour plus d'informations sur le lancement d'une instance Bottlerocket, consultez la section Lancer un Fusée à bouteilles instance pour Amazon ECS.

Lancer un Fusée à bouteilles instance pour Amazon ECS

Vous pouvez lancer une instance Bottlerocket de manière à pouvoir exécuter vos charges de travail de conteneur.

Vous pouvez utiliser le AWS CLI pour lancer le Bottlerocket instance.

1. Créez un fichier appelé `userdata.toml`. Ce fichier est utilisé pour les données utilisateur de l'instance. Remplacez *`cluster-name`* par le nom de votre cluster.

   ```
   [settings.ecs]
   cluster = "cluster-name"
   ```

2. Utilisez l'une des commandes incluses dans the section called "Récupération d'Amazon ECS-optimized Fusée à bouteilles Métadonnées AMI" pour obtenir l'ID d'AMI Bottlerocket. Utilisez ceci lors de l'étape suivante.
3. Exécutez la commande suivante pour lancer l'instance Bottlerocket. N'oubliez pas de remplacer les paramètres suivants :

   - Remplacer *`subnet`* avec l'ID du sous-réseau privé ou public dans lequel votre instance sera lancée.
   - Remplacer *`bottlerocket_ami`* à l'aide de l'ID AMI de l'étape précédente.
   - Remplacer *`t3.large`* avec le type d'instance que vous souhaitez utiliser.
   - Remplacer *`region`* avec le code de région.

```
aws ec2 run-instances --key-name ecs-bottlerocket-example \
   --subnet-id subnet \
   --image-id bottlerocket_ami \
   --instance-type t3.large \
   --region region \
   --tag-specifications
 'ResourceType=instance,Tags=[{Key=bottlerocket,Value=example}]' \
   --user-data file://userdata.toml \
   --iam-instance-profile Name=ecsInstanceRole
```

4. Exécutez la commande suivante pour vérifier que l'instance de conteneur est enregistrée auprès du cluster. Lorsque vous exécutez cette commande, n'oubliez pas de remplacer les paramètres suivants :

   - Remplacez *cluster* par le nom de votre cluster.
   - Remplacer *region* à l'aide de votre code de région.

```
aws ecs list-container-instances --cluster cluster-name --region region
```

Pour une présentation détaillée de la façon de démarrer avec le Bottlerocket système d'exploitation sur Amazon ECS, voir Utilisation d'une AMI Bottlerocket avec Amazon ECS sur GitHub et pour commencer Bottlerocket et Amazon ECS sur le AWS site de blog.

Gestion des instances de conteneur Amazon ECS Linux

Lorsque vous utilisez des instances EC2 pour vos charges de travail Amazon ECS, vous êtes responsable de la maintenance des instances

Procédures de gestion

- Lancement d'une instance de conteneur Amazon ECS Linux
- Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données
- Configuration des instances de conteneur Amazon ECS Linux pour recevoir des notifications relatives aux instances Spot
- Exécution d’un script lorsque vous lancez une instance de conteneur Amazon ECS Linux
- Augmentation des interfaces réseau d’une instance de conteneur Amazon ECS Linux

- Réservation de mémoire pour une instance de conteneur Amazon ECS Linux
- Gestion à distance des instances de conteneurs Amazon ECS à l'aide AWS Systems Manager
- Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux
- Configuration d'instances préinitialisées pour votre groupe Auto Scaling Amazon ECS
- Mise à jour de l'agent de conteneur Amazon ECS

Chaque version d'agent de conteneur Amazon ECS prend en charge un ensemble de différentes fonctions et fournit des correctifs des versions précédentes. Dans la mesure du possible, nous recommandons toujours d'utiliser la dernière version de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour voir quelles fonctionnalités et améliorations sont incluses dans chaque version de l'agent, consultez https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La version minimale de Docker pour des métriques fiables est la version Docker `v20.10.13` et versions plus récentes, incluses dans Amazon ECS-optimized AMI `20220607` et plus récents.
> La version de l'agent Amazon ECS `1.20.0` et les versions plus récentes ne prennent plus en charge les versions de Docker antérieures à `18.01.0`.

Lancement d'une instance de conteneur Amazon ECS Linux

Vos instances de conteneur Amazon ECS sont créées à l'aide de la console Amazon EC2.

Vous pouvez lancer une instance par différentes méthodes, notamment via la console Amazon EC2, AWS CLI, et le SDK. Pour plus d'informations sur les autres méthodes de lancement d'instance, consultez la section Lancer votre instance dans le Guide de l'utilisateur Amazon EC2.

Pour plus d'informations sur l'assistant de lancement, consultez la section Lancement d'une instance à l'aide du nouvel assistant de lancement d'instance dans le Guide de l'utilisateur Amazon EC2.

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

Vous pouvez utiliser le nouvel assistant Amazon EC2 pour lancer une instance. L'assistant de lancement d'instance spécifie les paramètres de lancement qui sont requis pour lancer une instance.

Paramètres pour la configuration d’instance

- Procédure
- Noms et identifications
- Images d’applications et de systèmes d’exploitation (Amazon Machine Image)
- Type d’instance
- Paire de clés (connexion)
- Paramètres réseau
- Configurer le stockage
- Détails avancés

Procédure

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

1. Ouvrez la console Amazon EC2 à l’adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation en haut de l'écran, l'actuel AWS La région est affichée (par exemple, USA Est (Ohio)). Sélectionnez une région dans laquelle lancer l’instance.
3. Sur le tableau de bord de la console Amazon EC2, sélectionnez Launch instance (Lancer une instance).

Noms et identifications

Le nom de l'instance est une identification, où la clé est Name (Nom), et la valeur est le nom que vous spécifiez. Vous pouvez étiqueter l'instance, les volumes et les Elastic Graphics. Pour les instances Spot, vous pouvez baliser uniquement la demande d’instance Spot.

La spécification d’un nom d’instance et d’identifications supplémentaires est facultative.

- Pour Name (Nom), saisissez un nom descriptif pour l’instance. Si vous ne spécifiez pas de nom, l’instance peut être identifiée par son ID, qui est automatiquement généré lorsque vous lancez l’instance.
- Pour ajouter des identifications supplémentaires, sélectionnez Add additional tags (Ajouter des identifications supplémentaires). Choisissez Add tag (Ajouter une identification), saisissez une clé et une valeur, puis sélectionnez le type de ressource à étiqueter. Choisissez Add tag (Ajouter une identification) pour chaque étiquette supplémentaire.

Images d’applications et de systèmes d’exploitation (Amazon Machine Image)

Une Amazon Machine Image (AMI) contient les informations requises pour créer une instance. Par exemple, une AMI peut contenir le logiciel nécessaire pour fonctionner en tant que serveur web, comme Apache et votre site web.

Utilisez le Rechercher barre pour trouver un Amazon approprié ECS-optimized AMI publié par AWS.

1. Entrez l'un des termes suivants dans la barre de recherche.
   - **`ami-ecs`**
   - Le Valeur d'une Amazon ECS-optimized AMI.

     Pour la dernière version d'Amazon ECS-optimized Les AMI et leurs valeurs, voir Amazon pour Linux ECS-optimized AMI.
2. Appuyez sur Entrée.
3. Sur la page Sélection d'une Amazon Machine Image (AMI) sélectionnez l'onglet AMI AWS Marketplace.
4. À partir du volet de gauche Refine results (Affiner les résultats), sélectionnez Amazon Web Services en tant qu'éditeur.
5. Choisissez Sélectionner sur la ligne de l'AMI que vous souhaitez utiliser.

   Vous pouvez également choisir Annuler pour revenir à l'assistant de lancement de l'instance sans choisir d'AMI. Une AMI par défaut sera sélectionnée. Assurez-vous que l'AMI répond aux exigences décrites dans Amazon ECS-optimized AMI Linux.

Type d’instance

Le type d’instance définit la configuration matérielle et la taille de l’instance. Les types d’instance plus importants disposent de plus d’UC et de mémoire. Pour plus d’informations, consultez Types d’instances dans le Guide de l’utilisateur Amazon EC2. Si vous souhaitez exécuter un IPv6-only charge de travail, certains types d'instances ne prennent pas en charge les adresses IPv6. Pour plus d'informations, consultez la section Adresses IPv6 dans le Guide de l'utilisateur Amazon EC2.

- Pour Instance type (Type d’instance), sélectionnez le type de l’instance.

  Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

Paire de clés (connexion)

Pour Key pair name (Nom de la paire de clés), choisissez une paire de clés existante ou choisissez Create new key pair (Créer une nouvelle paire de clés) pour en créer une nouvelle.

> **Important**
>
> Si vous sélectionnez l'option Proceed without key pair (Not recommended) ((Continuer sans paire de clé) (Non recommandé)), vous ne pourrez pas vous connecter à l'instance à moins de choisir une AMI configurée de façon à autoriser les utilisateurs à se connecter d'une autre façon.

Paramètres réseau

Configurez les paramètres réseau selon vos besoins après avoir cliqué sur le bouton Modifier dans la section Paramètres réseau du formulaire.

- Pour VPC, choisissez le VPC dans lequel vous souhaitez lancer l'instance. Pour exécuter un IPv6-only charge de travail, choisissez un VPC à double pile qui inclut à la fois un bloc d'adresse CIDR IPv4 et IPv6.
- Pour Sous-réseau, choisissez le sous-réseau dans lequel lancer l'instance. Vous pouvez lancer une instance dans un sous-réseau associé à une zone de disponibilité, une zone locale, une zone Wavelength ou un Outpost.

  Pour lancer l'instance dans une zone de disponibilité, sélectionnez le sous-réseau dans lequel lancer votre instance. Pour créer un sous-réseau, choisissez Créer un nouveau sous-réseau afin d'accéder à la console Amazon VPC. Une fois que vous avez terminé, revenez dans l'assistant de lancement d'instance et choisissez l'icône Refresh (Actualiser) afin de charger votre sous-réseau dans la liste.

  Pour lancer l'instance dans une zone locale, sélectionnez un sous-réseau que vous avez créé dans la zone locale.

  Pour lancer une instance dans un Outpost, sélectionnez un sous-réseau dans un VPC que vous avez associé à l'Outpost.

  Pour exécuter un IPv6-only charge de travail, choisissez un sous-réseau qui inclut uniquement un bloc d'adresse CIDR IPv6.

- Auto-assign IP publique: Si votre instance doit être accessible depuis Internet, vérifiez que Auto-assign IP publique le champ est défini sur Activer. Si ce n'est pas le cas, définissez ce champ sur Disable (Désactiver).

  > **Note**
  >
  > Les instances de conteneur ont besoin de communiquer avec le point de terminaison de service Amazon ECS. Cela peut être via un point de terminaison d'un VPC d'interface ou via vos instances de conteneur ayant des adresses IP publiques.
  > Pour plus d'informations sur les points de terminaison d'un VPC d'interface, consultez Points de terminaison VPC de l'interface Amazon ECS (AWS PrivateLink).
  > Si vous n'avez pas de point de terminaison d'un VPC d'interface configuré et que vos instances de conteneur n'ont pas d'adresses IP publiques, elles doivent utiliser la traduction d'adresse réseau (NAT) pour fournir cet accès. Pour de plus amples informations, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC et Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux dans ce guide.

- Si vous choisissez un VPC à double pile et IPv6-only sous-réseau, pour Auto-assign Adresse IP IPv6, choisissez Activer.
- Firewall (security groups) (Pare-feu (groupes de sécurité)) : utilisez un groupe de sécurité afin de définir les règles de pare-feu de votre instance de conteneur. Ces règles déterminent le trafic réseau entrant acheminé vers votre instance de conteneur. Le reste du trafic est ignoré.
  - Pour sélectionner un groupe de sécurité existant, choisissez Select existing security group (Sélectionner un groupe de sécurité existant), puis sélectionnez le groupe de sécurité que vous avez créé dans Configurer l'utilisation d'Amazon ECS.
- Si vous lancez l'instance pour un IPv6-only charge de travail, choisissez Configuration réseau avancée, puis pour Attribuer une adresse IP IPv6 principale, choisissez Oui.

  > **Note**
  >
  > Sans adresse IPv6 principale, les tâches exécutées sur l'instance de conteneur dans les modes réseau hôte ou pont échoueront à s'enregistrer auprès des équilibreurs de charge ou de AWS Cloud Map.

Configurer le stockage

L'AMI sélectionnée inclut un ou plusieurs volumes de stockage, notamment le volume racine. Vous pouvez spécifier d'autres volumes à attacher à l'instance.

Vous pouvez utiliser la vue Simple.

- Storage type (Type de stockage) : configurez le stockage pour votre instance de conteneur.

  Si vous utilisez Amazon ECS-optimized AMI Amazon Linux 2, votre instance possède un seul volume de 30 Go configuré, qui est partagé entre le système d'exploitation et Docker.

  Si vous utilisez Amazon ECS-optimized AMI, deux volumes sont configurés pour votre instance. Le volume Racine est réservé au système d'exploitation et le second volume Amazon EBS (attaché à `/dev/xvdcz`) est réservé à Docker.

  Vous pouvez augmenter ou diminuer la taille des volumes pour votre instance afin de répondre aux besoins de votre application.

Détails avancés

Développez la section Détails avancés pour afficher les champs et spécifier des paramètres supplémentaires pour l'instance.

- Purchasing option (Option d'achat) : sélectionnez Request Spot Instances (Demander des instances Spot) pour demander une instance Spot. Vous devez également définir les autres champs associés aux instances Spot. Pour plus d'informations, consultez Demandes d'instances Spot.

  > **Note**
  >
  > Si vous utilisez les instances Spot et qu'un message `Not available` s'affiche, vous devrez peut-être choisir un autre type d'instance.

- IAM instance profile (Profil d'instance IAM) : sélectionnez votre rôle IAM d'instance de conteneur. Celui-ci est généralement nommé `ecsInstanceRole`.

> **Important**
>
> Si vous ne pas lancez votre instance de conteneur avec les autorisations IAM appropriées, votre agent Amazon ECS ne peut pas se connecter à votre cluster. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

- Données utilisateur : configurez votre instance de conteneur Amazon ECS avec les données utilisateur, telles que les variables d'environnement d'agent depuis Configuration de l'agent de conteneur Amazon ECS. Les scripts de données utilisateur Amazon EC2 sont exécutés une seule fois, au premier lancement de l'instance. Les exemples suivants présentent des utilisations courantes des données utilisateur :
  - Par défaut, votre instance de conteneur est lancée dans votre cluster par défaut. Pour un lancement dans un cluster autre que celui défini par défaut, choisissez la liste Détails avancés. Collez ensuite le script suivant dans Données de l'utilisateur champ, remplacement *your_cluster_name* avec le nom de votre cluster.

    ```
    #!/bin/bash
    echo ECS_CLUSTER=your_cluster_name >> /etc/ecs/ecs.config
    ```

  - Si vous avez un fichier `ecs.config` dans Amazon S3 et que l'accès en lecture seule d'Amazon S3 à votre rôle d'instance de conteneur est activé, choisissez la liste Advanced Details (Détails avancés). Collez ensuite le script suivant dans Données de l'utilisateur champ. Remplacez *your-cluster-name* par le nom de votre cluster. Remplacer *amzn-s3-demo-bucket* avec le nom de votre bucket.

    > **Note**
    >
    > Pour en savoir plus sur cette configuration, consultez Stockage de la configuration d'instance de conteneur Amazon ECS dans Amazon S3.

    ```
    #!/bin/bash -e
    # Shut down after a user data failure.
    # Set InstanceInitiatedShutdownBehavior=terminate in the launch template to
     terminate the instance.
    trap 'status=$?; if [ "$status" -ne 0 ]; then shutdown -h now; fi' EXIT
    ```

```
# Set the cluster before any command that can fail.
echo ECS_CLUSTER=your-cluster-name >> /etc/ecs/ecs.config
yum install -y aws-cli
aws s3 cp s3://amzn-s3-demo-bucket/ecs.config /tmp/ecs.config
sed -i '/^ECS_CLUSTER=/d' /tmp/ecs.config
echo ECS_CLUSTER=your-cluster-name >> /tmp/ecs.config
install -m 0644 /tmp/ecs.config /etc/ecs/ecs.config
```

- Spécifiez des balises pour votre instance de conteneur à l'aide du paramètre de configuration `ECS_CONTAINER_INSTANCE_TAGS`. Cela permet de créer des balises associées uniquement à Amazon ECS, qui ne peuvent pas être répertoriées à l'aide de l'API Amazon EC2.

  > **Important**
  >
  > Si vous lancez vos instances de conteneur à l'aide d'un groupe Amazon EC2 Auto Scaling, vous devez utiliser le paramètre de configuration de l'agent ECS_CONTENER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux instances Amazon EC2 lancées à l'aide de groupes Auto Scaling.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=your_cluster_name
  ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
  EOF
  ```

- Spécifiez les balises pour votre instance de conteneur, puis utilisez le paramètre de configuration `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM` pour les propager depuis Amazon EC2 vers Amazon ECS.

  L'exemple suivant présente un script de données utilisateur qui propage les balises associées à une instance de conteneur, et enregistre l'instance de conteneur avec un cluster nommé `your_cluster_name` :

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=your_cluster_name
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

- Par défaut, l'agent de conteneur Amazon ECS essaie de détecter la compatibilité de l'instance de conteneur pour un IPv6-only configuration en examinant les routes IPv4 et IPv6 par défaut de l'instance. Pour annuler ce comportement, vous pouvez définir le paramètre `ECS_INSTANCE_IP_COMPATIBILITY` sur `ipv4` ou `ipv6` dans le fichier `/etc/ecs/ecs.config` de l'instance.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_INSTANCE_IP_COMPATIBILITY=ipv6
EOF
```

Pour plus d'informations, consultez Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données.

Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données

Lorsque vous lancez une instance Amazon EC2, vous pouvez transmettre les données utilisateur à cette instance. Ces données peuvent être utilisées afin d'effectuer des tâches de configuration automatisées courantes, voire d'exécuter des scripts lors du démarrage de l'instance. Pour Amazon ECS, les données utilisateur sont le plus souvent utilisées pour transmettre les informations de configuration au démon Docker et à l'agent de conteneur Amazon ECS.

Vous pouvez transmettre plusieurs types de données utilisateur à Amazon EC2, notamment des boothooks de cloud, des scripts shell et des directives `cloud-init`. Pour plus d'informations sur ces types de formats et sur d'autres types de formats, consultez Cloud-Init documentation.

Pour transmettre les données utilisateur lors de l'utilisation de l'assistant de lancement Amazon EC2, consultez la section Lancement d'une instance de conteneur Amazon ECS Linux.

Vous pouvez configurer l'instance de conteneur pour transmettre des données dans la configuration de l'agent de conteneur ou dans la configuration du démon Docker.

Agent de conteneur Amazon ECS

Les variantes Linux d'Amazon ECS-optimized AMI recherche les données de configuration de l'agent dans `/etc/ecs/ecs.config` fichier au démarrage de l'agent de conteneur. Vous pouvez spécifier ces données de configuration lors du lancement avec les données utilisateur Amazon EC2. Pour

plus d'informations sur les variables de configuration d'agent de conteneur Amazon ECS disponibles, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Pour définir une seule variable de configuration d'agent, par exemple le nom du cluster, utilisez echo afin de copier la variable dans le fichier de configuration :

```
#!/bin/bash
echo "ECS_CLUSTER=MyCluster" >> /etc/ecs/ecs.config
```

Si vous devez écrire plusieurs variables dans `/etc/ecs/ecs.config`, utilisez le format `heredoc` suivant. Ce format écrit tout entre les lignes commençant par cat et `EOF` dans le fichier de configuration.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
ECS_LOGLEVEL=debug
ECS_WARM_POOLS_CHECK=true
EOF
```

Pour définir des attributs d'instance personnalisés, définissez la variable d'environnement `ECS_INSTANCE_ATTRIBUTES`.

```
#!/bin/bash
cat <<'EOF' >> ecs.config
ECS_INSTANCE_ATTRIBUTES={"envtype":"prod"}
EOF
```

Démon Docker

Vous pouvez spécifier des informations de configuration du démon Docker avec les données utilisateur Amazon EC2. Pour plus d'informations sur les options de configuration , consultez la documentation sur le démon Docker.

> **Note**
>
> AWS ne prend pas en charge les configurations Docker personnalisées, car elles peuvent parfois entrer en conflit avec les futures modifications ou fonctionnalités d'Amazon ECS sans avertissement.

Dans l'exemple suivant, les options personnalisées sont ajoutées au fichier de configuration du démon Docker, `/etc/docker/daemon.json` qui est ensuite spécifiée dans les données utilisateur lors du lancement de l'instance.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"debug": true}
EOF
systemctl restart docker --no-block
```

Dans l'exemple suivant, les options personnalisées sont ajoutées au fichier de configuration du démon Docker, `/etc/docker/daemon.json` qui est ensuite spécifiée dans les données utilisateur lors du lancement de l'instance. Cet exemple montre comment désactiver Docker-proxy dans le fichier de configuration du démon Docker.

```
#!/bin/bash
cat <<EOF >/etc/docker/daemon.json
{"userland-proxy": false}
EOF
systemctl restart docker --no-block
```

Configuration des instances de conteneur Amazon ECS Linux pour recevoir des notifications relatives aux instances Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 fournit un avis d'interruption de deux minutes pour les actions de résiliation et d'arrêt des instances Spot. Il ne fournit pas l'avis de deux minutes pour l'action de mise en veille prolongée. Si le drainage des instances Spot Amazon ECS est activé sur l'instance, Amazon ECS reçoit l'avis d'interruption d'instance Spot et bascule l'instance à l'état `DRAINING`.

> **Important**
>
> Amazon ECS ne reçoit pas d'avis de la part d'Amazon EC2 lorsque les instances sont supprimées par le rééquilibrage de la capacité Auto Scaling. Pour plus d'informations, consultez Rééquilibrage de la capacité Amazon EC2 Auto Scaling.

Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur. Les tâches de service ayant l'état `PENDING` sur l'instance de conteneur faisant l'objet du drainage sont arrêtées immédiatement. S'il y a des instances de conteneur disponibles dans le cluster, des tâches de service de remplacement sont lancées dessus.

Le drainage des instances Spot est désactivé par défaut.

Vous pouvez activer le drainage d'instance Spot lorsque vous lancez une instance. Ajoutez le script suivant dans le champ Données utilisateur. Remplacer *`MyCluster`* avec le nom du cluster dans lequel enregistrer l'instance de conteneur.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
EOF
```

Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Pour activer le drainage des instances Spot pour une instance de conteneur existante

1. Connectez-vous à l'instance Spot via SSH.
2. Modifiez le fichier `/etc/ecs/ecs.config` et ajoutez le code suivant :

   ```
   ECS_ENABLE_SPOT_INSTANCE_DRAINING=true
   ```

3. Redémarrez le service `ecs`.

   - Pour l'Amazonie ECS-optimized AMI Amazon Linux 2 :

     ```
     sudo systemctl restart ecs
     ```

4. (Facultatif) Vous pouvez vérifier que l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur en interrogeant l'opération API d'introspection d'agent. Pour plus d'informations, consultez the section called "Introspection de conteneur".

   ```
   curl http://localhost:51678/v1/metadata
   ```

Exécution d'un script lorsque vous lancez une instance de conteneur Amazon ECS Linux

Vous devrez peut-être exécuter un conteneur spécifique sur chaque instance de conteneur pour traiter les problèmes de sécurité et de fonctionnement, telles que la surveillance, la sécurité, les métriques, la découverte de service ou la journalisation.

Pour ce faire, vous pouvez configurer vos instances de conteneur de façon à ce qu'elles appellent la commande docker run avec le script de données utilisateur au moment du lancement ou dans un système d'initialisation comme Upstart ou systemd. Bien que cette méthode soit efficace, elle présente quelques inconvénients, étant donné qu'Amazon ECS n'a pas connaissance du conteneur et qu'il ne peut pas surveiller l'UC, la mémoire, les ports ni aucune autre ressource utilisée. Afin de veiller à ce qu'Amazon ECS prenne en compte correctement toutes les ressources de la tâche, vous devez créer une définition de tâche pour le conteneur que vous voulez exécuter sur vos instances de conteneur. Ensuite, utilisez Amazon ECS pour placer la tâche au moment du lancement avec les données utilisateur Amazon EC2.

Le script de données utilisateur Amazon EC2 de la procédure suivante utilise l'API d'introspection Amazon ECS pour identifier l'instance de conteneur. Ensuite, il utilise le AWS CLI et le start-task commande pour exécuter une tâche spécifiée sur elle-même au démarrage.

Pour démarrer une tâche au moment du lancement d'une instance de conteneur

1. Modifiez votre rôle IAM `ecsInstanceRole` pour ajouter des autorisations pour l'opération d'API `StartTask`. Pour plus d'informations, consultez la section Mettre à jour les autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .
2. Lancez une ou plusieurs instances de conteneurs à l'aide d'Amazon ECS-optimized AMI Amazon Linux 2. Lancez de nouvelles instances de conteneur et utilisez l'exemple de script suivant dans les données utilisateur EC2. Remplacer *`your_cluster_name`* avec le cluster dans lequel l'instance de conteneur doit s'enregistrer et *`my_task_def`* avec la définition de la tâche à exécuter sur l'instance au lancement.

Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

> **Note**
>
> Le contenu MIME en plusieurs parties suivant utilise un script shell pour définir les valeurs de configuration et installer les packages. Il utilise également une tâche systemd pour démarrer la tâche après le lancement du service ecs et une fois que l'API d'introspection est disponible.

```
Content-Type: multipart/mixed; boundary="==BOUNDARY=="
MIME-Version: 1.0

--==BOUNDARY==
Content-Type: text/x-shellscript; charset="us-ascii"

#!/bin/bash
# Specify the cluster that the container instance should register into
cluster=your_cluster_name

# Write the cluster configuration variable to the ecs.config file
# (add any other configuration variables here also)
echo ECS_CLUSTER=$cluster >> /etc/ecs/ecs.config

START_TASK_SCRIPT_FILE="/etc/ecs/ecs-start-task.sh"
cat <<- 'EOF' > ${START_TASK_SCRIPT_FILE}
 exec 2>>/var/log/ecs/ecs-start-task.log
 set -x

 # Install prerequisite tools
 yum install -y jq aws-cli

 # Wait for the ECS service to be responsive
 until curl -s http://localhost:51678/v1/metadata
 do
  sleep 1
 done

 # Grab the container instance ARN and AWS Region from instance metadata
 instance_arn=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F/ '{print $NF}' )
```

```
 cluster=$(curl -s http://localhost:51678/v1/metadata | jq -r '. | .Cluster' | awk
 -F/ '{print $NF}' )
 region=$(curl -s http://localhost:51678/v1/metadata | jq -r '.
 | .ContainerInstanceArn' | awk -F: '{print $4}')

 # Specify the task definition to run at launch
 task_definition=my_task_def

 # Run the AWS CLI start-task command to start your task on this container instance
 aws ecs start-task --cluster $cluster --task-definition $task_definition --
container-instances $instance_arn --started-by $instance_arn --region $region
EOF

# Write systemd unit file
UNIT="ecs-start-task.service"
cat <<- EOF > /etc/systemd/system/${UNIT}
      [Unit]
      Description=ECS Start Task
      Requires=ecs.service
      After=ecs.service

      [Service]
      Restart=on-failure
      RestartSec=30
      ExecStart=/usr/bin/bash ${START_TASK_SCRIPT_FILE}

      [Install]
      WantedBy=default.target
EOF

# Enable our ecs.service dependent service with `--no-block` to prevent systemd
 deadlock
# See https://github.com/aws/amazon-ecs-agent/issues/1707
systemctl enable --now --no-block "${UNIT}"
----==BOUNDARY==--
```

3. Vérifiez que vos instances de conteneur sont lancées dans le cluster approprié et que vos tâches ont démarré.

   a. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.

   b. Dans la barre de navigation, sélectionnez la région dans laquelle se trouve votre cluster.

   c. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge vos instances de conteneur.

d. Sur la page Cluster, choisissez Tâches, puis choisissez vos tâches.

   Chaque instance de conteneur que vous avez lancée doit comporter votre tâche en cours d'exécution.

   Si vous ne voyez pas vos tâches, vous pouvez vous connecter à vos instances de conteneur avec le protocole SSH et vérifier le fichier `/var/log/ecs/ecs-start-task.log` pour consulter les informations de débogage.

Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux

> **Note**
>
> Cette fonction n'est pas disponible sur Fargate.

Les tâches qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'interfaces réseau qui peuvent être attachées à des instances Amazon EC2 est limité par défaut et l'interface réseau principale est considérée comme l'une d'elles. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois ENI qui lui sont attachées. L'interface réseau principale de l'instance est considérée comme une interface réseau. Vous pouvez donc attacher deux ENI supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une ENI, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance.

Amazon ECS prend en charge le lancement d'instances de conteneur avec une augmentation de la densité ENI et utilisant des types d'instances Amazon EC2 pris en charge. Lorsque vous utilisez ces types d'instances et activez le paramètre de compte `awsvpcTrunking`, des ENI supplémentaires sont disponibles sur les instances de conteneur nouvellement lancées. Cette configuration vous permet de placer plus de tâches sur chaque instance de conteneur. Pour utiliser la console afin d'activer la fonctionnalité, consultez la section Modification des paramètres de compte Amazon ECS. Pour utiliser le AWS CLI pour activer la fonction, voir Gérer les paramètres du compte Amazon ECS à l'aide du AWS CLI.

Par exemple, une instance `c5.large` dotée du paramètre `awsvpcTrunking` dispose d'une limite ENI augmentée à douze. L'instance de conteneur a alors une interface réseau principale. Amazon ECS crée et attache une interface réseau « tronc » à l'instance de conteneur. Par

conséquent, cette configuration vous permet de lancer dix tâches sur l'instance de conteneur au lieu des deux tâches actuellement possibles.

L'interface réseau « tronc » est entièrement gérée par Amazon ECS et est supprimée lorsque vous résiliez votre instance de conteneur ou lorsque vous annulez son enregistrement dans le cluster. Pour plus d'informations, consultez Options de mise en réseau des tâches Amazon ECS pour EC2.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez la fonctionnalité de jonction ENI.

- Uniquement les variantes Linux d'Amazon ECS-optimized AMI, ou autres variantes d'Amazon Linux avec version `1.28.1` ou version ultérieure de l'agent de conteneur et de la version `1.28.1-2` ou version ultérieure du package ecs-init, prenez en charge l'extension ENI limites. Si vous utilisez la dernière variante Linux d'Amazon ECS-optimized AMI, ces exigences seront satisfaites. Les conteneurs Windows ne sont pas pris en charge à l'heure actuelle.
- Seules les nouvelles instances Amazon EC2 lancées après l'activation du paramètre `awsvpcTrunking` reçoivent l'augmentation des limites ENI et l'interface réseau « de jonction ». Les instances lancées avant cette adoption ne reçoivent pas ces fonctionnalités, quelles que soient les actions réalisées.
- Les requêtes DNS IPv4 basées sur les ressources doivent être désactivées dans les instances Amazon EC2. Pour désactiver cette option, assurez-vous que l'option Activer les requêtes DNS IPV4 (registre A) basées sur les ressources est désélectionnée lors de la création d'une nouvelle instance à l'aide de la console Amazon EC2. Pour désactiver cette option à l'aide du AWS CLI, utilisez la commande suivante.

  ```
  aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-resource-name-dns-a-record --no-dry-run
  ```

- Les instances Amazon EC2 dans les sous-réseaux partagés ne sont pas prises en charge. Elles ne pourront pas s'enregistrer dans un cluster si elles sont utilisées.
- Vos tâches doivent utiliser le mode réseau `awsvpc` et l'EC2. Les tâches utilisant Fargate ont toujours reçu une ENI dédiée, quel que soit leur nombre. Cette fonctionnalité n'est donc pas nécessaire.
- Vos tâches doivent être lancées dans le même Amazon VPC que votre instance de conteneur. Vos tâches ne parviendront pas à démarrer avec une erreur d'attribut si elles ne sont pas dans le même VPC.

- Lors du lancement d'une nouvelle instance de conteneur, celle-ci passe à l'état `REGISTERING` tandis que l'Elastic Network Interface (ENI) « tronc » est mise en service pour l'instance. Si l'enregistrement échoue, l'instance passe à l'état `REGISTRATION_FAILED`. Vous pouvez dépanner un échec d'enregistrement en décrivant l'instance de conteneur pour afficher le champ `statusReason` qui décrit la raison de l'échec. L'instance de conteneur peut alors être désenregistrée ou résiliée manuellement. Une fois que l'instance de conteneur est correctement désenregistrée ou résiliée, Amazon ECS supprime l'ENI « tronc ».

  > **Note**
  >
  > Amazon ECS émet des événements de changement d'état d'instance de conteneur que vous pouvez contrôler pour les instances qui passent à un état `REGISTRATION_FAILED`. Pour plus d'informations, consultez Événements de changement d'état d'une instance de conteneur Amazon ECS.

- Une fois que l'instance de conteneur est résiliée, elle passe à l'état `DEREGISTERING` tandis que la mise en service de l'ENI « tronc » est annulée. L'instance passe alors à l'état `INACTIVE`.
- Si une instance de conteneur d'un sous-réseau public bénéficiant de l'augmentation des limites ENI est arrêtée puis redémarrée, elle perd son adresse IP publique et l'agent de conteneur perd sa connexion.
- Lorsque vous activez `awsvpcTrunking`, les instances de conteneur reçoivent une ENI supplémentaire qui utilise le groupe de sécurité par défaut du VPC et qui est gérée par Amazon ECS.

  Un VPC par défaut est fourni avec un sous-réseau public dans chaque zone de disponibilité, une passerelle Internet et des paramètres permettant la résolution DNS. Le sous-réseau par défaut est public, car la table de routage principale envoie le trafic du sous-réseau destiné à Internet vers la passerelle Internet. Vous pouvez transformer un sous-réseau par défaut en sous-réseau privé en supprimant l'itinéraire de la destination 0.0.0.0/0 vers la passerelle Internet. Toutefois, si vous procédez ainsi, aucune instance de conteneur exécutée dans ce sous-réseau ne pourra accéder à Internet. Vous pouvez ajouter ou supprimer des règles de groupe de sécurité pour contrôler le trafic entrant et sortant de vos sous-réseaux. Pour plus d'informations, consultez la section Règles des groupes de sécurité dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Conditions préalables

Avant de lancer une instance de conteneur bénéficiant d'une augmentation des limites ENI, vous devez veiller à ce que les conditions préalables suivantes soient réunies.

- Le rôle lié à un service pour Amazon ECS doit être créé. Le rôle lié au service Amazon ECS fournit à Amazon ECS les autorisations nécessaires pour passer des appels vers d'autres AWS services en votre nom. Ce rôle est généré automatiquement lorsque vous créez un cluster, ou si vous créez ou mettez à jour un service dans la Console de gestion AWS. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS. Vous pouvez également créer le rôle lié au service en procédant comme suit AWS CLI commande.

  ```
  aws iam create-service-linked-role --aws-service-name ecs.amazonaws.com
  ```

- Votre compte ou rôle IAM d'instance de conteneur doit activer le paramètre de compte `awsvpcTrunking`. Nous vous recommandons de créer deux rôles d'instance de conteneur (`ecsInstanceRole`). Vous pouvez ensuite activer le paramètre du compte `awsvpcTrunking` pour un rôle et utiliser ce rôle pour les tâches nécessitant une jonction ENI. Pour plus d'informations sur le rôle d'instance de conteneur, consultez la section Rôle IAM d'instance de conteneur Amazon ECS.

Une fois les conditions remplies, vous pouvez lancer une nouvelle instance de conteneur en utilisant l'un des types d'instances Amazon EC2 pris en charge. Cette instance bénéficiera de l'augmentation des limites ENI. Pour obtenir la liste des types d'instances, consultez Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS. L'instance de conteneur doit disposer de la version `1.28.1` ou ultérieure de l'agent de conteneur et de la version `1.28.1-2` ou ultérieure du package ecs-init. Si vous utilisez la dernière variante Linux d'Amazon ECS-optimized AMI, ces exigences seront satisfaites. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

> **Important**
>
> Les requêtes DNS IPv4 basées sur les ressources doivent être désactivées dans les instances Amazon EC2. Pour désactiver cette option, assurez-vous que l'option Activer les requêtes DNS IPV4 (registre A) basées sur les ressources est désélectionnée lors de la création d'une nouvelle instance à l'aide de la console Amazon EC2. Pour désactiver cette option à l'aide du AWS CLI, utilisez la commande suivante.

```
aws ec2 modify-private-dns-name-options --instance-id i-xxxxxxx --no-enable-
resource-name-dns-a-record --no-dry-run
```

Pour afficher vos instances de conteneurs avec une augmentation ENI limites avec AWS CLI

Chaque instance de conteneur possède une interface réseau par défaut, appelée interface réseau « tronc ». Utilisez la commande suivante pour répertorier vos instances de conteneur bénéficiant de l'augmentation des limites ENI en interrogeant l'attribut `ecs.awsvpc-trunk-id`, qui indique qu'elle dispose d'une interface réseau « de jonction ».

- list-attributes (AWS CLI)

```
aws ecs list-attributes \
      --target-type container-instance \
      --attribute-name ecs.awsvpc-trunk-id \
      --cluster cluster_name \
      --region us-east-1
```

- Get-ECSAttributeList (AWS Tools for Windows PowerShell)

```
Get-ECSAttributeList -TargetType container-instance -AttributeName ecs.awsvpc-trunk-
id -Region us-east-1
```

Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS

L'exemple suivant montre les types d'instances Amazon EC2 pris en charge et le nombre de tâches qui utilisent le mode réseau `awsvpc` pouvant être lancées sur chaque type d'instance avant et après l'activation du paramètre de compte `awsvpcTrunking`.

> **Important**
>
> Bien que d'autres types d'instance soient pris en charge dans la même famille d'instances, les types d'instance `a1.metal`, `c5.metal`, `c5a.8xlarge`, `c5ad.8xlarge`, `c5d.metal`, `m5.metal`, `p3dn.24xlarge`, `r5.metal`, `r5.8xlarge` et `r5d.metal` ne sont pas pris en charge.

Les familles d'instances `c5n`, `d3`, `d3en`, `g3`, `g3s` ,`g4dn`, `i3`, `i3en`, `inf1`, `m5dn`, `m5n`, `m5zn`, `mac1`, `r5b`, `r5n`, `r5dn`, `u-12tb1`, `u-6tb1`, `u-9tb1` et `z1d` ne sont pas prises en charge.

Rubriques

- Usage général
- Calcul optimisé
- Mémoire optimisée
- Stockage optimisé
- Calcul accéléré
- Calcul haute performance

Usage général

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
a1.medium	1	10
a1.large	2	10
a1.xlarge	3	20
a1.2xlarge	3	40
a1.4xlarge	7	60
m5.large	2	10
m5.xlarge	3	20
m5.2xlarge	3	40
m5.4xlarge	7	60
m5.8xlarge	7	60
m5.12xlarge	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m5.16xlarge	14	120
m5.24xlarge	14	120
m5a.large	2	10
m5a.xlarge	3	20
m5a.2xlarge	3	40
m5a.4xlarge	7	60
m5a.8xlarge	7	60
m5a.12xlarge	7	60
m5a.16xlarge	14	120
m5a.24xlarge	14	120
m5ad.large	2	10
m5ad.xlarge	3	20
m5ad.2xlarge	3	40
m5ad.4xlarge	7	60
m5ad.8xlarge	7	60
m5ad.12xlarge	7	60
m5ad.16xlarge	14	120
m5ad.24xlarge	14	120
m5d.large	2	10
m5d.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m5d.2xlarge	3	40
m5d.4xlarge	7	60
m5d.8xlarge	7	60
m5d.12xlarge	7	60
m5d.16xlarge	14	120
m5d.24xlarge	14	120
m5d.metal	14	120
m6a.large	2	10
m6a.xlarge	3	20
m6a.2xlarge	3	40
m6a.4xlarge	7	60
m6a.8xlarge	7	90
m6a.12xlarge	7	120
m6a.16xlarge	14	120
m6a.24xlarge	14	120
m6a.32xlarge	14	120
m6a.48xlarge	14	120
m6a.metal	14	120
m6g.medium	1	4
m6g.large	2	10

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m6g.xlarge	3	20
m6g.2xlarge	3	40
m6g.4xlarge	7	60
m6g.8xlarge	7	60
m6g.12xlarge	7	60
m6g.16xlarge	14	120
m6g.metal	14	120
m6gd.medium	1	4
m6gd.large	2	10
m6gd.xlarge	3	20
m6gd.2xlarge	3	40
m6gd.4xlarge	7	60
m6gd.8xlarge	7	60
m6gd.12xlarge	7	60
m6gd.16xlarge	14	120
m6gd.metal	14	120
m6i.large	2	10
m6i.xlarge	3	20
m6i.2xlarge	3	40
m6i.4xlarge	7	60

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m6i.8xlarge	7	90
m6i.12xlarge	7	120
m6i.16xlarge	14	120
m6i.24xlarge	14	120
m6i.32xlarge	14	120
m6i.metal	14	120
m6id.large	2	10
m6id.xlarge	3	20
m6id.2xlarge	3	40
m6id.4xlarge	7	60
m6id.8xlarge	7	90
m6id.12xlarge	7	120
m6id.16xlarge	14	120
m6id.24xlarge	14	120
m6id.32xlarge	14	120
m6id.metal	14	120
m6idn.large	2	10
m6idn.xlarge	3	20
m6idn.2xlarge	3	40
m6idn.4xlarge	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m6idn.8xlarge	7	90
m6idn.12xlarge	7	120
m6idn.16xlarge	14	120
m6idn.24xlarge	14	120
m6idn.32xlarge	15	120
m6idn.metal	15	120
m6in.large	2	10
m6in.xlarge	3	20
m6in.2xlarge	3	40
m6in.4xlarge	7	60
m6in.8xlarge	7	90
m6in.12xlarge	7	120
m6in.16xlarge	14	120
m6in.24xlarge	14	120
m6in.32xlarge	15	120
m6in.metal	15	120
m7a.medium	1	4
m7a.large	2	10
m7a.xlarge	3	20
m7a.2xlarge	3	40

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m7a.4xlarge	7	60
m7a.8xlarge	7	90
m7a.12xlarge	7	120
m7a.16xlarge	14	120
m7a.24xlarge	14	120
m7a.32xlarge	14	120
m7a.48xlarge	14	120
m7a.metal-48xl	14	120
m7g.medium	1	4
m7g.large	2	10
m7g.xlarge	3	20
m7g.2xlarge	3	40
m7g.4xlarge	7	60
m7g.8xlarge	7	60
m7g.12xlarge	7	60
m7g.16xlarge	14	120
m7g.metal	14	120
m7gd.medium	1	4
m7gd.large	2	10
m7gd.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m7gd.2xlarge	3	40
m7gd.4xlarge	7	60
m7gd.8xlarge	7	60
m7gd.12xlarge	7	60
m7gd.16xlarge	14	120
m7gd.metal	14	120
m7i.large	2	10
m7i.xlarge	3	20
m7i.2xlarge	3	40
m7i.4xlarge	7	60
m7i.8xlarge	7	90
m7i.12xlarge	7	120
m7i.16xlarge	14	120
m7i.24xlarge	14	120
m7i.48xlarge	14	120
m7i.metal-24xl	14	120
m7i.metal-48xl	14	120
m7i-flex.large	2	4
m7i-flex.xlarge	3	10
m7i-flex.2xlarge	3	20

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m7i-flex.4xlarge	7	40
m7i-flex.8xlarge	7	60
m7i-flex.12xlarge	7	120
m7i-flex.16xlarge	14	120
m8a.medium	1	4
m8a.large	2	10
m8a.xlarge	3	20
m 8a.2 x large	3	40
m8a.4xlarge	7	60
m 8 x 8 x large	9	90
m 8 A. 12 x large	11	120
m 8 A. 16 x large	15	120
m8a. 24 x large	15	120
m8a. 48 x large	23	120
m8a.metal-24xl	15	120
m8a.metal-48xl	23	120
m8azn.medium	2	4
m8azn.large	3	10
m8azn.xlarge	3	20
m8azn.3xlarge	7	40

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8 az.6 x large	7	60
m8 azn. 12 x large	15	120
m8 azn.24 x large	15	120
m8azn.metal-12xl	15	120
m8azn.metal-24xl	15	120
m8g.medium	1	4
m8g.large	2	10
m8g.xlarge	3	20
m8g.2xlarge	3	40
m8g.4xlarge	7	60
m8g.8xlarge	7	60
m8g.12xlarge	7	60
m8g.16xlarge	14	120
m8g.24xlarge	14	120
m8g.48xlarge	14	120
m8g.metal-24xl	14	120
m8g.metal-48xl	14	120
m8gb.medium	1	4
m8gb.large	2	10
m8gb.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
M8 Go. 2 x large	3	40
M8 Go. 4 x large	7	60
8 g x 8 x large	9	60
M8 Go. 12 x large	11	60
M8 Go. 16 x large	15	120
M8 Go, 24 x large	23	120
m 8 g x 48 x large	23	120
m8gb.metal-24xl	23	120
m8gb.metal-48xl	23	120
m8gd.medium	1	4
m8gd.large	2	10
m8gd.xlarge	3	20
m8gd.2xlarge	3	40
m8gd.4xlarge	7	60
m8gd.8xlarge	7	60
m8gd.12xlarge	7	60
m8gd.16xlarge	14	120
m8gd.24xlarge	14	120
m8gd.48xlarge	14	120
m8gd.metal-24xl	14	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8gd.metal-48xl	14	120
m8gn.medium	1	4
m8gn.large	2	10
m8gn.xlarge	3	20
m8 gn.2 x large	3	40
M8 Gn. 4 x large	7	60
8 g x 8 x large	9	60
8 g x 12 x large	11	60
8 g x 16 x large	15	120
8 g x 24 x large	23	120
8 g x 48 x large	23	120
m8gn.metal-24xl	23	120
m8gn.metal-48xl	23	120
m8i.large	2	10
m8i.xlarge	3	20
m8i.2xlarge	3	40
m8i.4xlarge	7	60
m8i.8xlarge	9	90
m8i.12xlarge	11	120
m8i.16xlarge	15	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8i.24xlarge	15	120
m8i.32xlarge	23	120
m8i.48xlarge	23	120
m8i.96xlarge	23	120
m8i.metal-48xl	23	120
m8i.metal-96xl	23	120
m8id.large	2	10
m8id.xlarge	3	20
m8 id.2 x large	3	40
m 8 x 4 x large	7	60
8 x x large	9	90
M8 id.12 x large	11	120
M8 x 16 x large	15	120
M8 id.24 x large	15	120
M8 id.32 x large	23	120
M8 x 48 x large	23	120
M8 x 96 x large	23	120
m8id.metal-48xl	23	120
m8id.metal-96xl	23	120
m8i-flex.large	2	4

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8i-flex.xlarge	3	10
m8i-flex.2xlarge	3	20
m8i-flex.4xlarge	7	40
m8i-flex.8xlarge	9	60
m8i-flex.12xlarge	11	120
m8i-flex.16xlarge	15	120
m 8 po de large	3	10
m8 pouces.large	3	20
m 8 po x 2 x large	3	40
8 po x 4 x large	7	60
8 pouces x large	7	90
m 8 po x 12 x large	11	120
m 8 po x 16 x large	15	120
m 8 po x 24 x large	15	120
m 8 po x 32 x large	15	120
m 8 po x 48 x large	23	120
m 8 po x 96 x large	23	120
8 po. Métal - 48 XL	23	120
8 po. Métal - 96 XL	23	120
m8idn.large	3	10

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8idn.xlarge	3	20
m8 idn.2 x large	3	40
m8 idn.4 x large	7	60
m8 idn.8 x large	7	90
m8 idn.12 x large	11	120
M8 idn.16 x large	15	120
M8 idn.24 x large	15	120
m8 idn.32 x large	15	120
M8 idn.48 x large	23	120
m8 idn.96 x large	23	120
m8idn.metal-48xl	23	120
m8idn.metal-96xl	23	120
m8ine.large	3	10
m8ine.xlarge	3	20
8 min. 2 x large	3	40
8 min.4 x large	7	60
8 m x 8 x large	7	90
8 min. 12 x large	11	120
m8 ib.large	3	10
m8ib.xlarge	3	20

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
8 m x 2 x large	3	40
8 m x 4 x large	7	60
8 x 8 x large	7	90
8 m x 12 x large	11	120
8 m x 16 x large	15	120
8 m x 24 x large	15	120
m 8 x 32 x large	15	120
m 8 x 48 x large	23	120
m 8 x 96 x large	23	120
m8ib.metal-48xl	23	120
m8ib.metal-96xl	23	120
m8idb.large	3	10
m8idb.xlarge	3	20
m8idb.2xlarge	3	40
m8idb.4xlarge	7	60
m8idb.8xlarge	7	90
m8 idb.12 x large	11	120
m8 idb.16 x large	15	120
m8 idb.24 x large	15	120
m8 idb.32 x large	15	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m8 idb.48 x large	23	120
m8idb.96xlarge	23	120
m8idb.metal-48xl	23	120
m8idb.metal-96xl	23	120
m9g.moyen	1	4
9 g. large	2	10
m9g.xlarge	3	20
9 g. 2 x large	3	40
9 g. 4 x large	7	60
9 m x 8 x large	9	60
9 g. 12 x large	11	60
9 g. 16 x large	15	120
9 g. 24 x large	23	120
9 g. 48 x large	23	120
m9g.metal-48xl	23	120
m9gd.medium	1	4
m9gd.large	2	10
m9gd.xlarge	3	20
m9gd.2 x large	3	40
m 9 g x 4 x large	7	60

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
m9 g x 8 x large	9	60
m9 g x 12 x large	11	60
m 9 g x 16 x large	15	120
m9 g x 24 x large	23	120
m 9 g x 48 x large	23	120
m9gd.metal-48xl	23	120
mac2.metal	7	12
mac2-m1ultra.metal	7	12
mac2-m2.metal	7	12
mac2-m2pro.metal	7	12
mac-m3 ultra métal	7	12
mac-m4.metal	7	12
mac-m4pro.metal	7	12
mac-m4max. métal	7	12
t8. Nano	1	2
t8i.micro	1	2
t8i.petit	2	4
t8i.moyen	2	4

Calcul optimisé

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c5.large	2	10
c5.xlarge	3	20
c5.2xlarge	3	40
c5.4xlarge	7	60
c5.9xlarge	7	60
c5.12xlarge	7	60
c5.18xlarge	14	120
c5.24xlarge	14	120
c5a.large	2	10
c5a.xlarge	3	20
c5a.2xlarge	3	40
c5a.4xlarge	7	60
c5a.12xlarge	7	60
c5a.16xlarge	14	120
c5a.24xlarge	14	120
c5ad.large	2	10
c5ad.xlarge	3	20
c5ad.2xlarge	3	40
c5ad.4xlarge	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c5ad.12xlarge	7	60
c5ad.16xlarge	14	120
c5ad.24xlarge	14	120
c5d.large	2	10
c5d.xlarge	3	20
c5d.2xlarge	3	40
c5d.4xlarge	7	60
c5d.9xlarge	7	60
c5d.12xlarge	7	60
c5d.18xlarge	14	120
c5d.24xlarge	14	120
c6a.large	2	10
c6a.xlarge	3	20
c6a.2xlarge	3	40
c6a.4xlarge	7	60
c6a.8xlarge	7	90
c6a.12xlarge	7	120
c6a.16xlarge	14	120
c6a.24xlarge	14	120
c6a.32xlarge	14	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c6a.48xlarge	14	120
c6a.metal	14	120
c6g.medium	1	4
c6g.large	2	10
c6g.xlarge	3	20
c6g.2xlarge	3	40
c6g.4xlarge	7	60
c6g.8xlarge	7	60
c6g.12xlarge	7	60
c6g.16xlarge	14	120
c6g.metal	14	120
c6gd.medium	1	4
c6gd.large	2	10
c6gd.xlarge	3	20
c6gd.2xlarge	3	40
c6gd.4xlarge	7	60
c6gd.8xlarge	7	60
c6gd.12xlarge	7	60
c6gd.16xlarge	14	120
c6gd.metal	14	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c6gn.medium	1	4
c6gn.large	2	10
c6gn.xlarge	3	20
c6gn.2xlarge	3	40
c6gn.4xlarge	7	60
c6gn.8xlarge	7	60
c6gn.12xlarge	7	60
c6gn.16xlarge	14	120
c6i.large	2	10
c6i.xlarge	3	20
c6i.2xlarge	3	40
c6i.4xlarge	7	60
c6i.8xlarge	7	90
c6i.12xlarge	7	120
c6i.16xlarge	14	120
c6i.24xlarge	14	120
c6i.32xlarge	14	120
c6i.metal	14	120
c6id.large	2	10
c6id.xlarge	3	20

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c6id.2xlarge	3	40
c6id.4xlarge	7	60
c6id.8xlarge	7	90
c6id.12xlarge	7	120
c6id.16xlarge	14	120
c6id.24xlarge	14	120
c6id.32xlarge	14	120
c6id.metal	14	120
c6in.large	2	10
c6in.xlarge	3	20
c6in.2xlarge	3	40
c6in.4xlarge	7	60
c6in.8xlarge	7	90
c6in.12xlarge	7	120
c6in.16xlarge	14	120
c6in.24xlarge	14	120
c6in.32xlarge	15	120
c6in.metal	15	120
c7a.medium	1	4
c7a.large	2	10

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c7a.xlarge	3	20
c7a.2xlarge	3	40
c7a.4xlarge	7	60
c7a.8xlarge	7	90
c7a.12xlarge	7	120
c7a.16xlarge	14	120
c7a.24xlarge	14	120
c7a.32xlarge	14	120
c7a.48xlarge	14	120
c7a.metal-48xl	14	120
c7g.medium	1	4
c7g.large	2	10
c7g.xlarge	3	20
c7g.2xlarge	3	40
c7g.4xlarge	7	60
c7g.8xlarge	7	60
c7g.12xlarge	7	60
c7g.16xlarge	14	120
c7g.metal	14	120
c7gd.medium	1	4

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c7gd.large	2	10
c7gd.xlarge	3	20
c7gd.2xlarge	3	40
c7gd.4xlarge	7	60
c7gd.8xlarge	7	60
c7gd.12xlarge	7	60
c7gd.16xlarge	14	120
c7gd.metal	14	120
c7gn.medium	1	4
c7gn.large	2	10
c7gn.xlarge	3	20
c7gn.2xlarge	3	40
c7gn.4xlarge	7	60
c7gn.8xlarge	7	60
c7gn.12xlarge	7	60
c7gn.16xlarge	14	120
c7gn.metal	14	120
c7i.large	2	10
c7i.xlarge	3	20
c7i.2xlarge	3	40

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c7i.4xlarge	7	60
c7i.8xlarge	7	90
c7i.12xlarge	7	120
c7i.16xlarge	14	120
c7i.24xlarge	14	120
c7i.48xlarge	14	120
c7i.metal-24xl	14	120
c7i.metal-48xl	14	120
c7i-flex.large	2	4
c7i-flex.xlarge	3	10
c7i-flex.2xlarge	3	20
c7i-flex.4xlarge	7	40
c7i-flex.8xlarge	7	60
c7i-flex.12xlarge	7	120
c7i-flex.16xlarge	14	120
c8a.moyen	1	4
c8a.large	2	10
c8a.xlarge	3	20
environ 8 x 2 x large	3	40
environ 8 x 4 x large	7	60

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
environ 8 x 8 x large	9	90
environ 8 x 12 x large	11	120
environ 8 x 16 x large	15	120
environ 8 x 24 x large	15	120
environ 8 x 48 x large	23	120
c8a.metal-24xl	15	120
c8a.metal-48xl	23	120
c8g.medium	1	4
c8g.large	2	10
c8g.xlarge	3	20
c8g.2xlarge	3	40
c8g.4xlarge	7	60
c8g.8xlarge	7	60
c8g.12xlarge	7	60
c8g.16xlarge	14	120
c8g.24xlarge	14	120
c8g.48xlarge	14	120
c8g.metal-24xl	14	120
c8g.metal-48xl	14	120
c8gb.medium	1	4

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c8gb.large	2	10
c8gb.xlarge	3	20
C8 Go. 2 x large	3	40
C8 Go. 4 x large	7	60
8 Go x 8 x large	9	60
C8 Go. 12 x large	11	60
C8 Go. 16 x large	15	120
C8 Go. 24 x large	23	120
C8 Go. 48 x large	23	120
c8gb.metal-24xl	23	120
c8gb.metal-48xl	23	120
c8gd.medium	1	4
c8gd.large	2	10
c8gd.xlarge	3	20
c8gd.2xlarge	3	40
c8gd.4xlarge	7	60
c8gd.8xlarge	7	60
c8gd.12xlarge	7	60
c8gd.16xlarge	14	120
c8gd.24xlarge	14	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c8gd.48xlarge	14	120
c8gd.metal-24xl	14	120
c8gd.metal-48xl	14	120
c8gn.medium	1	4
c8gn.large	2	10
c8gn.xlarge	3	20
c8gn.2xlarge	3	40
c8gn.4xlarge	7	60
c8gn.8xlarge	9	60
c8gn.12xlarge	11	60
c8gn.16xlarge	15	120
c8gn.24xlarge	23	120
c8gn.48xlarge	23	120
c8gn.metal-24xl	23	120
c8gn.metal-48xl	23	120
c8i.large	2	10
c8i.xlarge	3	20
c8i.2xlarge	3	40
c8i.4xlarge	7	60
environ 8 x 8 x large	9	90

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c8 x 12 x large	11	120
c8 x 16 x large	15	120
c8i.24 x large	15	120
c8 x 32 x large	23	120
c8 x 48 x large	23	120
c8i.96xlarge	23	120
c8i.metal-48xl	23	120
c8i.metal-96xl	23	120
c8id.large	2	10
c8id.xlarge	3	20
C8 id.2 x large	3	40
C8 id.4 x large	7	60
8 x x large	9	90
C8 id.12 x large	11	120
C8 id.16 x large	15	120
C8 id.24 x large	15	120
C8 id.32 x large	23	120
C8 id.48 x large	23	120
C8 id.96 x large	23	120
c8id.metal-48xl	23	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c8id.metal-96xl	23	120
c8i-flex.large	2	4
c8i-flex.xlarge	3	10
c8i-flex.2xlarge	3	20
c8i-flex.4xlarge	7	40
c8i-flex.8xlarge	9	60
c8i-flex.12xlarge	11	120
c8i-flex.16xlarge	15	120
8 pouces de large	3	10
c8in.xlarge	3	20
8 po x 2 x large	3	40
8 po x 4 x large	7	60
8 pouces x large	7	90
8 po x 12 x large	11	120
8 po x 16 x large	15	120
8 po x 24 x large	15	120
8 po x 32 x large	15	120
8 po x 48 x large	23	120
8 pouces x 96 x large	23	120
8 po. Métal - 48 XL	23	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
8 po. Métal - 96 XL	23	120
c8ine.large	3	10
c8ine.xlarge	3	20
8 pouces. 2 x large	3	40
8 pouces x large	7	60
8 pouces x large	7	90
8 pouces. 12 x large	11	120
c8 ib.large	3	10
c8ib.xlarge	3	20
C8 en 2 x large	3	40
8 pouces x 4 x large	7	60
8 pouces x 8 x large	7	90
8 pouces x 12 x large	11	120
8 pouces x 16 x large	15	120
8 pouces x 24 x large	15	120
8 pouces x 32 x large	15	120
8 pouces x 48 x large	23	120
8 pouces, 96 x large	23	120
c8ib.metal-48xl	23	120
c8ib.metal-96xl	23	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c9g.medium	1	4
C9 g. large	2	10
c9g.xlarge	3	20
9 cm x 2 x large	3	40
9 cm x 4 x large	7	60
9 cm x 8 x large	9	60
9 g. 12 x large	11	60
9 g. 16 x large	15	120
9 g. 24 x large	23	120
c9 g. 48 x large	23	120
c9g.metal-48xl	23	120
c9gd.medium	1	4
c9gd.large	2	10
c9gd.xlarge	3	20
c9gd.2xlarge	3	40
c9gd.4xlarge	7	60
9 x x 8 x large	9	60
C9 g x 12 x large	11	60
C9 x 16 x large	15	120
c9 g x 24 x large	23	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
c9 g x 48 x large	23	120
c9gd.metal-48xl	23	120

Mémoire optimisée

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r5.large	2	10
r5.xlarge	3	20
r5.2xlarge	3	40
r5.4xlarge	7	60
r5.12xlarge	7	60
r5.16xlarge	14	120
r5.24xlarge	14	120
r5a.large	2	10
r5a.xlarge	3	20
r5a.2xlarge	3	40
r5a.4xlarge	7	60
r5a.8xlarge	7	60
r5a.12xlarge	7	60
r5a.16xlarge	14	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r5a.24xlarge	14	120
r5ad.large	2	10
r5ad.xlarge	3	20
r5ad.2xlarge	3	40
r5ad.4xlarge	7	60
r5ad.8xlarge	7	60
r5ad.12xlarge	7	60
r5ad.16xlarge	14	120
r5ad.24xlarge	14	120
r5b.16xlarge	14	120
r5d.large	2	10
r5d.xlarge	3	20
r5d.2xlarge	3	40
r5d.4xlarge	7	60
r5d.8xlarge	7	60
r5d.12xlarge	7	60
r5d.16xlarge	14	120
r5d.24xlarge	14	120
r5dn.16xlarge	14	120
r6a.large	2	10

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r6a.xlarge	3	20
r6a.2xlarge	3	40
r6a.4xlarge	7	60
r6a.8xlarge	7	90
r6a.12xlarge	7	120
r6a.16xlarge	14	120
r6a.24xlarge	14	120
r6a.32xlarge	14	120
r6a.48xlarge	14	120
r6a.metal	14	120
r6g.medium	1	4
r6g.large	2	10
r6g.xlarge	3	20
r6g.2xlarge	3	40
r6g.4xlarge	7	60
r6g.8xlarge	7	60
r6g.12xlarge	7	60
r6g.16xlarge	14	120
r6g.metal	14	120
r6gd.medium	1	4

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r6gd.large	2	10
r6gd.xlarge	3	20
r6gd.2xlarge	3	40
r6gd.4xlarge	7	60
r6gd.8xlarge	7	60
r6gd.12xlarge	7	60
r6gd.16xlarge	14	120
r6gd.metal	14	120
r6i.large	2	10
r6i.xlarge	3	20
r6i.2xlarge	3	40
r6i.4xlarge	7	60
r6i.8xlarge	7	90
r6i.12xlarge	7	120
r6i.16xlarge	14	120
r6i.24xlarge	14	120
r6i.32xlarge	14	120
r6i.metal	14	120
r6id.large	2	10
r6id.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r6id.2xlarge	3	40
r6id.4xlarge	7	60
r6id.8xlarge	7	90
r6id.12xlarge	7	120
r6id.16xlarge	14	120
r6id.24xlarge	14	120
r6id.32xlarge	14	120
r6id.metal	14	120
r6idn.large	2	10
r6idn.xlarge	3	20
r6idn.2xlarge	3	40
r6idn.4xlarge	7	60
r6idn.8xlarge	7	90
r6idn.12xlarge	7	120
r6idn.16xlarge	14	120
r6idn.24xlarge	14	120
r6idn.32xlarge	15	120
r6idn.metal	15	120
r6in.large	2	10
r6in.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r6in.2xlarge	3	40
r6in.4xlarge	7	60
r6in.8xlarge	7	90
r6in.12xlarge	7	120
r6in.16xlarge	14	120
r6in.24xlarge	14	120
r6in.32xlarge	15	120
r6in.metal	15	120
r7a.medium	1	4
r7a.large	2	10
r7a.xlarge	3	20
r7a.2xlarge	3	40
r7a.4xlarge	7	60
r7a.8xlarge	7	90
r7a.12xlarge	7	120
r7a.16xlarge	14	120
r7a.24xlarge	14	120
r7a.32xlarge	14	120
r7a.48xlarge	14	120
r7a.metal-48xl	14	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r7g.medium	1	4
r7g.large	2	10
r7g.xlarge	3	20
r7g.2xlarge	3	40
r7g.4xlarge	7	60
r7g.8xlarge	7	60
r7g.12xlarge	7	60
r7g.16xlarge	14	120
r7g.metal	14	120
r7gd.medium	1	4
r7gd.large	2	10
r7gd.xlarge	3	20
r7gd.2xlarge	3	40
r7gd.4xlarge	7	60
r7gd.8xlarge	7	60
r7gd.12xlarge	7	60
r7gd.16xlarge	14	120
r7gd.metal	14	120
r7i.large	2	10
r7i.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r7i.2xlarge	3	40
r7i.4xlarge	7	60
r7i.8xlarge	7	90
r7i.12xlarge	7	120
r7i.16xlarge	14	120
r7i.24xlarge	14	120
r7i.48xlarge	14	120
r7i.metal-24xl	14	120
r7i.metal-48xl	14	120
r7iz.large	2	10
r7iz.xlarge	3	20
r7iz.2xlarge	3	40
r7iz.4xlarge	7	60
r7iz.8xlarge	7	90
r7iz.12xlarge	7	120
r7iz.16xlarge	14	120
r7iz.32xlarge	14	120
r7iz.metal-16xl	14	120
r7iz.metal-32xl	14	120
r8a.medium	1	4

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r8a.large	2	10
r8a.xlarge	3	20
r8a.2xlarge	3	40
r8a.4xlarge	7	60
r8a.8xlarge	9	90
r 8A. 12 x large	11	120
r8a. 16 x large	15	120
r8a. 24 x large	15	120
r8a. 48 x large	23	120
r8a.metal-24xl	15	120
r8a.metal-48xl	23	120
r8g.medium	1	4
r8g.large	2	10
r8g.xlarge	3	20
r8g.2xlarge	3	40
r8g.4xlarge	7	60
r8g.8xlarge	7	60
r8g.12xlarge	7	60
r8g.16xlarge	14	120
r8g.24xlarge	14	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r8g.48xlarge	14	120
r8g.metal-24xl	14	120
r8g.metal-48xl	14	120
r8gb.medium	1	4
r8gb.large	2	10
r8gb.xlarge	3	20
r8gb.2xlarge	3	40
r8gb.4xlarge	7	60
r8gb.8xlarge	9	60
r8gb.12xlarge	11	60
r8gb.16xlarge	15	120
r8gb.24xlarge	23	120
r8 g x 48 x large	23	120
r8gb.metal-24xl	23	120
r8gb.metal-48xl	23	120
r8gd.medium	1	4
r8gd.large	2	10
r8gd.xlarge	3	20
r8gd.2xlarge	3	40
r8gd.4xlarge	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r8gd.8xlarge	7	60
r8gd.12xlarge	7	60
r8gd.16xlarge	14	120
r8gd.24xlarge	14	120
r8gd.48xlarge	14	120
r8gd.metal-24xl	14	120
r8gd.metal-48xl	14	120
r8gn.medium	1	4
r8gn.large	2	10
r8gn.xlarge	3	20
r8gn.2xlarge	3	40
r8gn.4xlarge	7	60
r8gn.8xlarge	9	60
r8gn.12xlarge	11	60
r8gn.16xlarge	15	120
r8gn.24xlarge	23	120
r8gn.48xlarge	23	120
r8gn.metal-24xl	23	120
r8gn.metal-48xl	23	120
r8i.large	2	10

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r8i.xlarge	3	20
r8i.2xlarge	3	40
r8i.4xlarge	7	60
r8i.8xlarge	9	90
r8i.12xlarge	11	120
r8i.16xlarge	15	120
r8i.24xlarge	15	120
r8i.32xlarge	23	120
r8i.48xlarge	23	120
r8i.96xlarge	23	120
r8i.metal-48xl	23	120
r8i.metal-96xl	23	120
R8id.large	2	10
r8id.xlarge	3	20
R8 id.2 x large	3	40
R8 x 4 x large	7	60
8 x x large	9	90
R8 id.12 x large	11	120
R8 id.16 x large	15	120
R8 id.24 x large	15	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
R8 id.32 x large	23	120
R8 x 48 x large	23	120
R8 id.96 x large	23	120
R8id.metal-48xl	23	120
R8id.Metal-96XL	23	120
r8i-flex.large	2	4
r8i-flex.xlarge	3	10
r8i-flex.2xlarge	3	20
r8i-flex.4xlarge	7	40
r8i-flex.8xlarge	9	60
r8i-flex.12xlarge	11	120
r8i-flex.16xlarge	15	120
8 pouces de large	3	10
r8in.xlarge	3	20
8 pouces, 2 x large	3	40
8 pouces x large	7	60
8 pouces x large	7	90
8 po x 12 x large	11	120
8 po x 16 x large	15	120
8 po x 24 x large	15	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
8 po x 32 x large	15	120
8 po x 48 x large	23	120
8 po x 96 x large	23	120
8 po. Métal - 48 XL	23	120
8 po. Métal - 96 XL	23	120
r8idn.large	3	10
r8idn.xlarge	3	20
r8 idn.2 x large	3	40
r8 idn.4 x large	7	60
r8 idn.8 x large	7	90
r8 idn.12 x large	11	120
r8 idn.16 x large	15	120
r8 idn.24 x large	15	120
r8 idn.32 x large	15	120
r8 idn.48 x large	23	120
r8 idn.96 x large	23	120
R8idn.metal-48xl	23	120
R8idn.metal-96xl	23	120
R8 ib.large	3	10
r8ib.xlarge	3	20

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
R8 ib.2 x large	3	40
R8 en 4 x large	7	60
8 pouces x 8 x large	7	90
R8 ib.12 x large	11	120
R8 ib.16 x large	15	120
R8 ib.24 x large	15	120
R8 ib.32 x large	15	120
R8 ib.48 x large	23	120
R8 ib.96 x large	23	120
R8ib.metal-48xl	23	120
R8ib.metal-96xl	23	120
r8idb.large	3	10
r8idb.xlarge	3	20
r8idb.2xlarge	3	40
r8idb.4xlarge	7	60
r8idb.8xlarge	7	90
r8 idb.12 x large	11	120
r8 idb.16 x large	15	120
r8 idb.24 x large	15	120
r8 idb.32 x large	15	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
r8 idb.48 x large	23	120
r8idb.96xlarge	23	120
r8idb.metal-48xl	23	120
r8idb.metal-96xl	23	120
r9g. Medium	1	4
R9g. Taille L	2	10
r9g.xlarge	3	20
9 g x 2 x large	3	40
R9 g. 4 x large	7	60
9 g x 8 x large	9	60
9 g. 12 x large	11	60
R9 g. 16 x large	15	120
R9 g. 24 x large	23	120
R9 g. 48 x large	23	120
r9g.metal-48xl	23	120
r9gd.medium	1	4
r9gd.large	2	10
r9gd.xlarge	3	20
r9gd.2xlarge	3	40
R9 g x 4 x large	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
R9 x 8 x large	9	60
R9 g x 12 x large	11	60
R9 x 16 x large	15	120
R9 g x 24 x large	23	120
R9 g x 48 x large	23	120
r9gd.metal-48xl	23	120
u-3tb1.56xlarge	7	12
u-6tb1.56xlarge	14	12
u-18tb1.112xlarge	14	12
u-18tb1.metal	14	12
u-24tb1.112xlarge	14	12
u-24tb1.metal	14	12
u7i-6tb.112xlarge	14	120
u7i-8tb.112xlarge	14	120
u7i-12tb.224xlarge	14	120
u7in-16tb.224xlarge	15	120
u7in-24tb.224xlarge	15	120
u7in-32tb.224xlarge	15	120
u7inh-32tb.480xlarge	15	120
x2gd.medium	1	10

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
x2gd.large	2	10
x2gd.xlarge	3	20
x2gd.2xlarge	3	40
x2gd.4xlarge	7	60
x2gd.8xlarge	7	60
x2gd.12xlarge	7	60
x2gd.16xlarge	14	120
x2gd.metal	14	120
x2idn.16xlarge	14	120
x2idn.24xlarge	14	120
x2idn.32xlarge	14	120
x2idn.metal	14	120
x2iedn.xlarge	3	13
x2iedn.2xlarge	3	29
x2iedn.4xlarge	7	60
x2iedn.8xlarge	7	120
x2iedn.16xlarge	14	120
x2iedn.24xlarge	14	120
x2iedn.32xlarge	14	120
x2iedn.metal	14	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
x2iezn.2xlarge	3	64
x2iezn.4xlarge	7	120
x2iezn.6xlarge	7	120
x2iezn.8xlarge	7	120
x2iezn.12xlarge	14	120
x2iezn.metal	14	120
x8g.medium	1	4
x8g.large	2	10
x8g.xlarge	3	20
x8g.2xlarge	3	40
x8g.4xlarge	7	60
x8g.8xlarge	7	60
x8g.12xlarge	7	60
x8g.16xlarge	14	120
x8g.24xlarge	14	120
x8g.48xlarge	14	120
x8g.metal-24xl	14	120
x8g.metal-48xl	14	120
x8aedz.large	3	10
x8aedz.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
x 8 aedz. 3 x large	7	40
x 8 aedz. 6 x large	7	60
x 8 aedz. 12 x large	15	120
x 8 aedz. 24 x large	15	120
x8aedz.metal-12xl	15	120
x8aedz.metal-24xl	15	120
x8i.large	2	10
x8i.xlarge	3	20
x 8 x 2 x large	3	40
x 8 x 4 x large	7	60
x 8 x 8 x large	9	90
x 8 x 1 x 12 x large	11	120
x 8 x 1 x 16 x large	15	120
x 8 x 24 x large	15	120
x 8 x 32 x large	23	120
x 8 x 48 x large	23	120
x 8 x 64 x large	23	120
x 8 x 9,6 x large	23	120
x8i.metal-48xl	23	120
x8i.metal-96xl	23	120

Stockage optimisé

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
i4g.large	2	10
i4g.xlarge	3	20
i4g.2xlarge	3	40
i4g.4xlarge	7	60
i4g.8xlarge	7	60
i4g.16xlarge	14	120
i4i.xlarge	3	8
i4i.2xlarge	3	28
i4i.4xlarge	7	58
i4i.8xlarge	7	118
i4i.12xlarge	7	118
i4i.16xlarge	14	248
i4i.24xlarge	14	118
i4i.32xlarge	14	498
i4i.metal	14	498
i7i.large	2	10
i7i.xlarge	3	20
i7i.2xlarge	3	40
i7i.4xlarge	7	60

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
i7i.8xlarge	7	90
i7i.12xlarge	7	90
i7i.16xlarge	14	120
i7i.24xlarge	14	120
i7i.48xlarge	14	120
i7i.metal-24xl	14	120
i7i.metal-48xl	14	120
i7ie.large	2	20
i7ie.xlarge	3	29
i7ie.2xlarge	3	29
i7ie.3xlarge	3	29
i7ie.6xlarge	7	60
i7ie.12xlarge	7	60
i7ie.18xlarge	14	120
i7ie.24xlarge	14	120
i7ie.48xlarge	14	120
i7ie.metal-24xl	14	120
i7ie.metal-48xl	14	120
i8g.large	2	10
i8g.xlarge	3	20

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
i8g.2xlarge	3	40
i8g.4xlarge	7	60
i8g.8xlarge	7	60
i8g.12xlarge	7	60
i8g.16xlarge	14	120
i8g.24xlarge	14	120
i8g.48xlarge	14	120
i8g.metal-24xl	14	120
i8g.metal-48xl	14	120
i8ge.large	2	20
i8ge.xlarge	3	29
i8ge.2xlarge	3	29
i8ge.3xlarge	5	29
i8ge.6xlarge	9	60
i8ge.12xlarge	11	60
i8ge.18xlarge	15	120
i8ge.24xlarge	15	120
i8ge.48xlarge	23	120
i8ge.metal-24xl	15	120
i8ge.metal-48xl	23	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
im4gn.large	2	10
im4gn.xlarge	3	20
im4gn.2xlarge	3	40
im4gn.4xlarge	7	60
im4gn.8xlarge	7	60
im4gn.16xlarge	14	120
is4gen.medium	1	4
is4gen.large	2	10
is4gen.xlarge	3	20
is4gen.2xlarge	3	40
is4gen.4xlarge	7	60
is4gen.8xlarge	7	60

Calcul accéléré

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
dl1.24xlarge	59	120
dl2q.24xlarge	14	120
f2.6xlarge	7	90
f2.12xlarge	7	120

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
f2.48xlarge	14	120
g4ad.xlarge	1	12
g4ad.2xlarge	1	12
g4ad.4xlarge	2	12
g4ad.8xlarge	3	12
g4ad.16xlarge	7	12
g5.xlarge	3	6
g5.2xlarge	3	19
g5.4xlarge	7	40
g5.8xlarge	7	90
g5.12xlarge	14	120
g5.16xlarge	7	120
g5.24xlarge	14	120
g5.48xlarge	6	120
g5g.xlarge	3	20
g5g.2xlarge	3	40
g5g.4xlarge	7	60
g5g.8xlarge	7	60
g5g.16xlarge	14	120
g5g.metal	14	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
g6.xlarge	3	20
g6.2xlarge	3	40
g6.4xlarge	7	60
g6.8xlarge	7	90
g6.12xlarge	7	120
g6.16xlarge	14	120
g6.24xlarge	14	120
g6.48xlarge	14	120
g6e.xlarge	3	20
g6e.2xlarge	3	40
g6e.4xlarge	7	60
g6e.8xlarge	7	90
g6e.12xlarge	9	120
g6e.16xlarge	14	120
g6e.24xlarge	19	120
g6e.48xlarge	39	120
g6f.large	1	10
g6f.xlarge	3	20
g6f.2xlarge	3	40
g6f.4xlarge	7	60

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
gr6.4xlarge	7	60
gr6.8xlarge	7	90
gr6f.4xlarge	7	60
g 7,2 x large	3	40
g7,4 x large	7	60
g7,8 x large	9	90
g 7,12 x large	11	120
g 7,24 x large	23	120
g 7,48 x large	23	120
g7e.2 x large	3	242
g7,4 x large	7	242
g7,8 x large	7	242
g7e x 12 x large	9	242
g7e x 24 x large	19	242
g7e.48 x large	39	242
inf2.xlarge	3	20
inf2.8xlarge	7	90
inf2.24xlarge	14	120
inf2.48xlarge	14	120
p4d.24xlarge	59	120

Type d’instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
p4de.24xlarge	59	120
p5.4xlarge	3	60
p5.48xlarge	63	242
p5e.48xlarge	63	242
p5en.48xlarge	63	242
p6-b200.48xlarge	31	242
p6-b 300,48 x large	67	242
p6e-gb200.36xlarge	38	120
trn1.2xlarge	3	19
trn1.32xlarge	39	120
trn1n.32xlarge	79	242
trn 2,3 x large	1	14
trn2.48xlarge	31	242
trn2u.48xlarge	31	242
vt1.3xlarge	3	40
vt1.6xlarge	7	60
vt1.24xlarge	14	120

Calcul haute performance

Type d'instance	Limite de tâche sans jonction ENI	Limite de tâche avec jonction ENI
hpc6a.48xlarge	1	120
hpc6id.32xlarge	1	120
hpc7g.4xlarge	3	120
hpc7g.8xlarge	3	120
hpc7g.16xlarge	3	120
hpc8a.96x large	3	-2

Réservation de mémoire pour une instance de conteneur Amazon ECS Linux

Lorsque l'agent de conteneur Amazon ECS enregistre une instance de conteneur dans un cluster, il doit déterminer la quantité de mémoire disponible que l'instance de conteneur peut réserver pour vos tâches. En raison de la surcharge de mémoire de la plateforme et de la mémoire utilisée par le noyau du système, ce nombre est différent de la quantité de mémoire installée qui est annoncée pour les instances Amazon EC2. Par exemple, une instance `m4.large` a 8 Gio de mémoire installée. Toutefois, cela n'équivaut pas toujours exactement à 8 192 Mio de mémoire disponible pour les tâches lorsque l'instance de conteneur s'enregistre.

Détermination des ressources mémoire des instances gérées ECS

Amazon ECS Managed Instances utilise une approche hiérarchique pour déterminer les besoins en ressources mémoire pour les tâches. Contrairement à ECS sur EC2 qui repose sur l'introspection de la mémoire de Docker, ECS Managed Instances calcule les besoins en mémoire directement à partir de la charge utile des tâches lors des décisions de planification.

Lorsque l'agent ECS Managed Instances reçoit une tâche, il calcule la mémoire requise selon l'ordre de priorité suivant :

1. Task-level mémoire (priorité la plus élevée) - Si la mémoire au niveau des tâches est spécifiée dans la définition de la tâche, l'agent utilise directement cette valeur. Cela a priorité sur tous les paramètres de mémoire au niveau du conteneur.

2. Container-level somme de la mémoire (solution de secours) - Si la mémoire au niveau de la tâche n'est pas spécifiée (ou vaut 0), l'agent additionne les besoins en mémoire de tous les conteneurs de la tâche. Pour chaque contenant, il utilise :
   a. Réservation de mémoire (limite souple) - Si un conteneur le spécifie `memoryReservation` dans sa configuration, l'agent utilise cette valeur.
   b. Mémoire du conteneur (limite stricte) - Si `memoryReservation` n'est pas spécifié, l'agent utilise le contenant `memory` champ.

Example Task-level mémoire spécifiée

Lorsque la mémoire au niveau des tâches est spécifiée, elle a priorité sur les paramètres au niveau du conteneur :

```
{
  "family": "my-task",
  "memory": "2048",
  "containerDefinitions": [
    {
      "name": "container1",
      "memory": 1024,
      "memoryReservation": 512
    }
  ]
}
```

L'agent réserve 2 048 Mo (la mémoire au niveau des tâches est prioritaire).

Example Container-level mémoire avec réserves

Lorsque la mémoire au niveau des tâches n'est pas spécifiée, l'agent additionne les besoins en mémoire du conteneur :

```
{
  "family": "my-task",
  "containerDefinitions": [
    {
      "name": "container1",
      "memory": 1024,
      "memoryReservation": 512
    },
```

```
    {
      "name": "container2",
      "memory": 512
    }
  ]
}
```

L'agent réserve 512 MiB (réservation conteneur1) + 512 MiB (mémoire conteneur2) = 1024 MiB au total.

L'agent ECS Managed Instances effectue le calcul de la mémoire en trois phases :

1. Réception des tâches - Lorsqu'une charge utile de tâche arrive depuis le plan de contrôle ECS, l'agent calcule immédiatement la mémoire requise.
2. Stockage des ressources - La mémoire requise calculée est stockée dans le modèle de tâches pour une utilisation ultérieure dans les opérations de comptabilité des ressources.
3. Décision de planification - Avant d'accepter une tâche, l'agent vérifie si la mémoire disponible est suffisante. Si la mémoire disponible est insuffisante, la tâche est rejetée et reste dans la file d'attente du service ECS jusqu'à ce que les ressources soient disponibles.

> **Note**
>
> Contrairement à ECS sur EC2, ECS Managed Instances n'utilise pas `ECS_RESERVED_MEMORY` variable de configuration. La réservation de mémoire pour les processus du système est gérée par le biais de la gestion des ressources de la plateforme sous-jacente, et l'agent effectue une comptabilité précise des ressources en fonction des définitions de tâches.

Pour ECS sur EC2, l'agent de conteneur Amazon ECS fournit une variable de configuration appelée `ECS_RESERVED_MEMORY`, que vous pouvez utiliser pour supprimer un certain nombre de Mo de mémoire du pool alloué à vos tâches. Cela réserve de manière effective cette quantité de mémoire pour les processus système critiques.

Si vous occupez la totalité de la mémoire sur une instance de conteneur avec vos tâches, il est alors possible que vos tâches se disputent la mémoire avec les processus système critiques, ce qui peut provoquer une défaillance du système.

Par exemple, si vous spécifiez `ECS_RESERVED_MEMORY=256` dans votre fichier de configuration de l'agent de conteneur, l'agent enregistre la mémoire totale, moins 256 Mio pour cette instance. 256 Mio de mémoire ne sont donc pas allouées par des tâches ECS. Pour plus d'informations sur les variables de configuration de l'agent et la façon de les définir, consultez Configuration de l'agent de conteneur Amazon ECS et Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données.

Si vous spécifiez 8 192 Mio pour la tâche et qu'aucune de vos instances de conteneur ne dispose d'au moins 8 192 Mio de mémoire disponible pour répondre à cette exigence, la tâche ne peut pas être placée dans votre cluster. Si vous utilisez un environnement informatique géré, AWS Batch doit lancer un type d'instance plus grand pour répondre à la demande.

L'agent de conteneur Amazon ECS utilise la fonction `ReadMemInfo()` Docker pour connaître la mémoire totale disponible pour le système d'exploitation. Linux et Windows fournissent tous deux des utilitaires de ligne de commande pour déterminer la mémoire totale.

Example- Déterminer la mémoire totale Linux

La commande free renvoie la mémoire totale reconnue par le système d'exploitation.

```
$ free -b
```

Exemple de sortie pour un `m4.large` instance exécutant Amazon ECS-optimized AMI Amazon Linux.

```
             total       used       free     shared    buffers     cached
Mem:    8373026816  348180480 8024846336      90112   25534464  205418496
-/+ buffers/cache:  117227520 8255799296
```

Cette instance comporte 8373026816 octets de mémoire totale, soit 7 985 Mio disponible pour les tâches.

Example- Déterminer la mémoire totale Windows

La commande wmic renvoie la mémoire totale reconnue par le système d'exploitation.

```
C:\> wmic ComputerSystem get TotalPhysicalMemory
```

Exemple de sortie pour un `m4.large` instance exécutant Amazon ECS-optimized AMI Windows Server.

```
TotalPhysicalMemory
8589524992
```

Cette instance comporte 8589524992 octets de mémoire totale, soit 8 191 Mio disponible pour les tâches.

Affichage de la mémoire d'une instance de conteneur

Vous pouvez consulter la quantité de mémoire avec laquelle une instance de conteneur s'enregistre dans la console Amazon ECS (ou avec le DescribeContainerInstances Fonctionnement de l'API). Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, vous pouvez observer la mémoire disponible pour cette instance de conteneur, puis attribuer autant de mémoire à vos tâches.

Pour afficher la mémoire d'une instance de conteneur

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge votre instance de conteneur.
3. Choisissez Infrastructure, puis sous les instances de conteneur, choisissez une instance de conteneur.
4. La section Ressources indique la mémoire enregistrée et la mémoire disponible pour l'instance de conteneur.

   La valeur de mémoire Enregistrée correspond à l'instance de conteneur enregistrée auprès d'Amazon ECS lors de son premier lancement, tandis que la valeur de mémoire Disponible correspond à la mémoire qui n'a pas encore été allouée à des tâches.

Gestion à distance des instances de conteneurs Amazon ECS à l'aide AWS Systems Manager

Vous pouvez utiliser la fonction Exécuter la commande dans AWS Systems Manager (Systems Manager) pour gérer à distance et en toute sécurité la configuration de vos instances de conteneurs Amazon ECS. Exécutez la commande vous fournit des outils simples afin d'effectuer des tâches administratives courantes sans qu'il soit nécessaire de se connecter localement à l'instance. Vous pouvez gérer les changements de configuration sur vos clusters en exécutant des commandes simultanément sur plusieurs instances de conteneur. La fonctionnalité Exécuter la commande affiche le statut et les résultats de chaque commande.

Voici quelques exemples des types de tâches susceptibles d'être effectués avec la fonctionnalité Exécuter la commande :

- Installer ou désinstaller des packages
- Effectuer des mises à jour de sécurité
- Nettoyer les images Docker
- Arrêter ou démarrer des services
- Afficher les ressources système
- Afficher les fichiers journaux
- Effectuer des opérations sur les fichiers

Pour plus d'informations sur Run Command, veuillez consulter AWS Systems Manager Run Command dans le Guide de l'utilisateur AWS Systems Manager .

Les conditions suivantes sont requises pour utiliser Systems Manager avec Amazon ECS.

1. Vous devez attribuer le rôle d'instance de conteneur (ecsInstanceRole) autorisations pour accéder aux API de Systems Manager. Vous pouvez le faire en attribuant AmazonSSMManagedInstanceCore à la `ecsInstanceRole` rôle. Pour plus d’informations sur la manière d’associer une politique à un rôle, consultez la section Modification des autorisations pour un rôle dans le Guide de l’utilisateur Gestion des identités et des accès AWS .
2. Vérifier que SSM Agent est installé sur vos instances de conteneur. Pour plus d’informations, consultez la section Installation et désinstallation manuelle de l’agent SSM sur les instances EC2 pour Linux.

Une fois que vous avez associé les politiques gérées de Systems Manager à votre `ecsInstanceRole` et vérifiez que AWS Systems Manager L'agent (agent SSM) est installé sur vos instances de conteneur. Vous pouvez commencer à utiliser Run Command pour envoyer des commandes à vos instances de conteneur. Pour de plus amples informations sur l'exécution des commandes et des scripts shell sur vos instances et pour afficher la sortie obtenue, veuillez consulter Exécution de commandes avec Systems Manager Run Command et Procédures Run Command dans le Guide de l'utilisateur AWS Systems Manager .

Un cas d’utilisation courant consiste à mettre à jour le logiciel d’instance de conteneur avec Exécuter une commande. Vous pouvez suivre les procédures décrites dans AWS Systems Manager Guide de l'utilisateur avec les paramètres suivants.

Paramètre	Value
Document de commande	`AWS-RunShellScript`
Commande	`$ yum update -y`
Instances cibles	Vos instances de conteneur

Utilisation d’un proxy HTTP pour les instances de conteneur Amazon ECS Linux

Vous pouvez configurer vos instances de conteneur Amazon ECS pour qu'elles utilisent un proxy HTTP pour l'agent de conteneur Amazon ECS et le démon Docker. C'est utile si vos instances de conteneur n'ont pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

Pour configurer votre instance de conteneur Linux Amazon ECS afin qu'elle utilise un proxy HTTP, définissez les variables suivantes dans les fichiers appropriés lors du lancement (avec les données utilisateur Amazon EC2). Vous pouvez également modifier manuellement le fichier de configuration, puis redémarrer l’agent.

`/etc/ecs/ecs.config` (Amazon Linux 2 et AmazonLinux AMI)

`HTTP_PROXY=10.0.0.131:3128`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que l'agent Amazon ECS puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

`NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock`

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance du proxy.

`/etc/systemd/system/ecs.service.d/http-proxy.conf` (Amazon Linux 2 uniquement)

`Environment="HTTP_PROXY=10.0.0.131:3128/"`

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que `ecs-init` puisse se connecter à Internet. Par exemple, vos

instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2,/var/run/
docker.sock"
```

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance du proxy.

`/etc/init/ecs.override` (Amazon Linux AMI uniquement)

```
env HTTP_PROXY=10.0.0.131:3128
```

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que `ecs-init` puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
env NO_PROXY=169.254.169.254,169.254.170.2,/var/run/docker.sock
```

Utilisez la valeur `169.254.169.254,169.254.170.2,/var/run/docker.sock` pour filtrer des métadonnées d'instance EC2, les rôles IAM et le trafic du démon Docker en provenance du proxy.

`/etc/systemd/system/docker.service.d/http-proxy.conf` (Amazon Linux 2 uniquement)

```
Environment="HTTP_PROXY=http://10.0.0.131:3128"
```

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que le démon Docker puisse se connecter à Internet. Par exemple, vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
Environment="NO_PROXY=169.254.169.254,169.254.170.2"
```

Utilisez la valeur `169.254.169.254,169.254.170.2` pour filtrer les métadonnées d'instance EC2 issues du proxy.

`/etc/sysconfig/docker` (Amazon Linux AMI et Amazon Linux 2 uniquement)

```
export HTTP_PROXY=http://10.0.0.131:3128
```

Pour cette valeur, employez le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que le démon Docker puisse se connecter à Internet. Par exemple,

vos instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
export NO_PROXY=169.254.169.254,169.254.170.2
```

Utilisez la valeur `169.254.169.254,169.254.170.2` pour filtrer les métadonnées d'instance EC2 issues du proxy.

La définition de ces variables d'environnement dans les fichiers précédents n'affecte que l'agent de conteneur Amazon ECS, `ecs-init`, et le démon Docker. Elles ne configurent aucun autre service (comme yum) pour utiliser le proxy.

Pour plus d'informations sur la configuration du proxy, voir Comment configurer un proxy HTTP pour Docker et l'agent de conteneur Amazon ECS dans Amazon Linux 2 ou AL2023.

Configuration d’instances préinitialisées pour votre groupe Auto Scaling Amazon ECS

Amazon ECS prend en charge les groupes d'instances pré-initialisées Amazon EC2 Auto Scaling. Un groupe d'instances pré-initialisées est un groupe d'instances Amazon EC2 pré-initialisées prêtes à être mises en service. Chaque fois que votre application doit monter en puissance, Amazon EC2 Auto Scaling utilise les instances pré-initialisées du groupe d'instances pré-initialisées plutôt que de lancer des instances froides, permet l'exécution de tout processus d'initialisation final, puis met l'instance en service.

Pour en savoir plus sur les groupes instances pré-initialisées et comment les ajouter à votre groupe Auto Scaling, consultez la section Groupes d'instances pré-initialisées pour Amazon EC2 Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling.

Lorsque vous créez ou mettez à jour un groupe instances pré-initialisées pour un groupe Auto Scaling pour Amazon ECS, vous ne pouvez pas définir l'option qui renvoie les instances vers le groupe instances pré-initialisées à mise à l'échelle horizontale (`ReuseOnScaleIn`). Pour plus d'informations, consultez put-warm-pool dans la Référence AWS Command Line Interface .

Pour utiliser des groupes d'instances pré-initialisées avec votre cluster Amazon ECS, définissez la variable de configuration de l'agent `ECS_WARM_POOLS_CHECK` sur `true` dans le champ User data (Données utilisateur) du modèle de lancement de votre groupe Amazon EC2 Auto Scaling.

L'exemple suivant montre comment la variable de configuration de l'agent peut être spécifiée dans le Données de l'utilisateur champ d'un modèle de lancement Amazon EC2 pour les instances

Linux. Remplacez *MyCluster* par le nom de votre cluster. Pour les instances Windows, consultez Utilisation de pools chauds avec les instances Windows d'Amazon ECS.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_WARM_POOLS_CHECK=true
EOF
```

La variable ECS_WARM_POOLS_CHECK est uniquement prise en charge sur les versions d'agent 1.59.0 et ultérieures. Pour plus d'informations sur la variable, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.

Utilisation de pools chauds avec les instances Windows d'Amazon ECS

Lorsque vous utilisez des piscines chaudes avec Arrêté état du pool sur les instances Windows, utilisez les hooks de cycle de vie Amazon EC2 Auto Scaling pour vous assurer que l'exécution des données utilisateur est terminée avant l'arrêt de l'instance. Sans hooks de cycle de vie, l'instance peut être arrêtée avant la fin de l'initialisation d'Amazon ECS.

Pour configurer les hooks de cycle de vie pour les instances Windows Warm Pool, procédez comme suit :

1. Ajoutez la politique IAM suivante à votre rôle d'instance de conteneur pour permettre à l'instance de terminer l'action du cycle de vie :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "autoscaling:CompleteLifecycleAction"
            ],
            "Resource": "*"
        }
    ]
}
```

2. Ajoutez un hook de cycle de vie à votre groupe Auto Scaling sur Lancement de l'instance avec ABANDON comme résultat par défaut et un délai d'expiration du rythme cardiaque qui laisse

suffisamment de temps pour l'initialisation d'Amazon ECS (par exemple, 10 minutes). Pour plus d'informations, consultez la rubrique Utilisation de hooks de cycle de vie du Guide de l'utilisateur Amazon EC2 Auto Scaling.

3. Utilisez les données utilisateur suivantes pour créer une tâche planifiée afin de terminer l'action du cycle de vie après un certain temps. Ce délai garantit qu'EC2Launch termine d'enregistrer son état avant l'arrêt de l'instance. Remplacer *MyCluster*, *my-lifecycle-hook*, et *my-asg-name* avec tes valeurs.

```
<powershell>
$lifecycleHookName = "my-lifecycle-hook"
$autoScalingGroupName = "my-asg-name"

$scriptContent = @'
$token = Invoke-RestMethod -Method Put -Uri http://169.254.169.254/latest/api/token -
Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"}
$instanceId = Invoke-RestMethod -Method Get -Uri http://169.254.169.254/latest/meta-
data/instance-id -Headers @{"X-aws-ec2-metadata-token" = $token}
$region = Invoke-RestMethod -Method Get -Uri http://169.254.169.254/latest/meta-data/
placement/region -Headers @{"X-aws-ec2-metadata-token" = $token}
$lifecycleHookName = "my-lifecycle-hook"
$autoScalingGroupName = "my-asg-name"
Complete-ASLifecycleAction -LifecycleHookName $lifecycleHookName -
AutoScalingGroupName $autoScalingGroupName -InstanceId $instanceId -
LifecycleActionResult 'CONTINUE' -Region $region
'@
Set-Content -Path "C:\ProgramData\Amazon\ECS\complete-lifecycle.ps1" -Value
 $scriptContent

try
{
    [Environment]::SetEnvironmentVariable("ECS_WARM_POOLS_CHECK", "true", "Machine")
    Import-Module ECSTools
    Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole

    $taskAction = New-ScheduledTaskAction -Execute "powershell.exe" -Argument
 "-NoProfile -ExecutionPolicy Bypass -File C:\ProgramData\Amazon\ECS\complete-
lifecycle.ps1"
    $taskTriggerOnce = New-ScheduledTaskTrigger -Once -At ((Get-Date).AddSeconds(60))
    $taskTriggerStartup = New-ScheduledTaskTrigger -AtStartup
    $taskTriggerStartup.Delay = "PT60S"
    $settings = New-ScheduledTaskSettingsSet -StartWhenAvailable
```

```
    $taskPrincipal = New-ScheduledTaskPrincipal -UserId "SYSTEM" -LogonType
 ServiceAccount -RunLevel Highest
    Register-ScheduledTask -TaskName "CompleteLifecycleHook" -Action $taskAction -
Trigger $taskTriggerOnce, $taskTriggerStartup -Principal $taskPrincipal -Settings
 $settings -Force
}
catch
{
    $token = Invoke-RestMethod -Method Put -Uri http://169.254.169.254/latest/api/
token -Headers @{"X-aws-ec2-metadata-token-ttl-seconds" = "21600"}
    $INSTANCE = Invoke-RestMethod -Method Get -Uri http://169.254.169.254/latest/
meta-data/instance-id -Headers @{"X-aws-ec2-metadata-token" = $token}
    $region = Invoke-RestMethod -Method Get -Uri http://169.254.169.254/latest/meta-
data/placement/region -Headers @{"X-aws-ec2-metadata-token" = $token}
    Complete-ASLifecycleAction -InstanceId $INSTANCE -LifecycleHookName
 $lifecycleHookName -AutoScalingGroupName $autoScalingGroupName -
LifecycleActionResult ABANDON -Region $region
}
</powershell>
```

Ce script de données utilisateur effectue les opérations suivantes :

- Initialise l'agent Amazon ECS avec la vérification des piscines chaudes activée.
- Crée une tâche planifiée qui attend 60 secondes après l'achèvement des données utilisateur, puis signale au hook du cycle de vie de continuer.
- Crée une tâche planifiée qui s'exécute à chaque redémarrage suivant et signale au hook du cycle de vie de se poursuivre au bout de 60 secondes.
- Si l'initialisation échoue, l'action du cycle de vie est terminée avec ABANDON l'instance est donc arrêtée et remplacée.

Mise à jour de l'agent de conteneur Amazon ECS

L’agent de conteneur Amazon ECS doit parfois être mis à jour afin d’obtenir les correctifs de bogues et les nouvelles fonctions. La mise à jour de l'agent de conteneur Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution sur l'instance de conteneur. Le processus de mise à jour de l'agent varie selon que votre instance de conteneur a été lancée avec un Amazon ECS-optimized AMI ou autre système d'exploitation.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Vérification de la version d'agent de conteneur Amazon ECS

Vous pouvez vérifier la version de l'agent de conteneur qui est en cours d'exécution sur vos instances de conteneur afin de savoir s'il doit être mis à jour. L'affichage d'une instance de conteneur dans la console Amazon ECS fournit la version de l'agent. Procédez comme indiqué ci-dessous pour vérifier la version de votre agent.

Amazon ECS console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance externe.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet.
5. Sous Container instances (Instances de conteneur), notez la colonne Agent version (Version d'agent) pour vos instances de conteneur. Si l'instance de conteneur ne contient pas la dernière version de l'agent de conteneur, la console vous prévient à l'aide d'un message et signale la version d'agent obsolète.

   Si votre version d'agent est obsolète, vous pouvez la mettre à jour en suivant les instructions ci-dessous :

   - Si votre instance de conteneur exécute un Amazon ECS-optimized AMI, voir Mettre à jour l'agent de conteneur Amazon ECS sur un Amazon ECS-optimized AMI.
   - Si votre instance de conteneur n'exécute pas un Amazon ECS-optimized AMI, voir Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-Optimized AMI).

> **Important**
>
> Pour mettre à jour la version de l'agent Amazon ECS à partir de versions antérieures à la v1.0.0 sur votre Amazon ECS-optimized AMI, nous vous recommandons de mettre fin à votre instance de conteneur actuelle et d'en lancer une nouvelle avec la version la plus récente de l'AMI. Toutes les instances de conteneur qui utilisent une version précédente doivent être retirées et remplacées par l'AMI la plus récente. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Amazon ECS container agent introspection API

Vous pouvez également utiliser l'API d'introspection d'agent de conteneur Amazon ECS pour vérifier la version de l'agent à partir de l'instance de conteneur elle-même. Pour plus d'informations, consultez Introspection de conteneur Amazon ECS.

Pour vérifier si votre agent de conteneur Amazon ECS exécute la dernière version avec l'API d'introspection

1. Connectez-vous à votre instance de conteneur via SSH.
2. Interrogez l'API d'introspection.

```
[ec2-user ~]$ curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

> **Note**
>
> L'API d'introspection a ajouté les informations `Version` dans la version v1.0.0 de l'agent de conteneur Amazon ECS. Si `Version` n'apparaît pas lors de l'interrogation de l'API d'introspection ou que l'API d'introspection n'apparaît pas du tout dans votre agent, la version que vous exécutez est v0.0.3 ou une version antérieure. Vous devez mettre à jour votre version.

Mettre à jour l'agent de conteneur Amazon ECS sur un Amazon ECS-optimized AMI

Si vous utilisez un Amazon ECS-optimized AMI, plusieurs options s'offrent à vous pour obtenir la dernière version de l'agent de conteneur Amazon ECS (présentée par ordre de recommandation) :

- Mettez fin à l'instance de conteneur et lancez la dernière version d'Amazon ECS-optimized AMI Amazon Linux 2 (manuellement ou en mettant à jour votre configuration de lancement Auto Scaling avec la dernière AMI). Cette opération fournit une instance de conteneur propre avec les versions les plus récemment testées et validées d'Amazon Linux, de Docker, d'`ecs-init` et de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.
- Connectez-vous à l'instance avec SSH et mettez à jour le package `ecs-init` (et ses dépendances) vers la dernière version. Cette opération fournit les versions les plus récemment testées et validées de Docker et d'`ecs-init` disponibles dans les référentiels Amazon Linux et la dernière version de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Pour mettre à jour le `ecs-init` colis sur Amazon ECS-optimized AMI.
- Mettez à jour l'agent de conteneur avec `UpdateContainerAgent` Fonctionnement de l'API, soit via la console, soit avec AWS CLI ou AWS SDK. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS avec l'opération d'API `UpdateContainerAgent`.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Pour mettre à jour le **`ecs-init`** colis sur Amazon ECS-optimized AMI

1. Connectez-vous à votre instance de conteneur via SSH.
2. Mettez à jour le package `ecs-init` avec la commande suivante.

```
sudo yum update -y ecs-init
```

> **Note**
>
> Le package `ecs-init` et l'agent de conteneur Amazon ECS sont mis à jour immédiatement. Cependant, les versions les plus récentes de Docker ne sont pas

chargées tant que le démon Docker n'est pas redémarré. Redémarrez en relançant l'instance ou en exécutant les commandes suivantes sur votre instance :

- Amazon ECS-optimized AMI Amazon Linux 2 :

```
sudo systemctl restart docker
```

- Amazon ECS-optimized AMI Amazon Linux :

```
sudo service docker restart && sudo start ecs
```

Mise à jour de l'agent de conteneur Amazon ECS avec l'opération d'API **`UpdateContainerAgent`**

> **Important**
>
> Le `UpdateContainerAgent` L'API n'est prise en charge que sur les variantes Linux d'Amazon ECS-optimized AMI, à l'exception de l'Amazon ECS-optimized AMI Amazon Linux 2 (arm64). Pour les instances de conteneurs utilisant Amazon ECS-optimized AMI Amazon Linux 2 (arm64), mettez à jour `ecs-init` package pour mettre à jour l'agent. Pour les instances de conteneur qui exécutent d'autres systèmes d'exploitation, consultez la page Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-Optimized AMI). Si vous utilisez des instances de conteneur Windows, nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Le `UpdateContainerAgent` Le processus d'API commence lorsque vous demandez une mise à jour de l'agent, soit via la console, soit à l'aide du AWS CLI ou AWS SDK. Amazon ECS vérifie la version actuelle de votre agent par rapport à la dernière version disponible, et détermine si une mise à jour est possible. Si aucune mise à jour n'est disponible, si l'agent exécute déjà par exemple la version la plus récente, le message `NoUpdateAvailableException` est renvoyé.

Les étapes du processus de mise à jour sont les suivantes :

`PENDING`

Une mise à jour de l'agent est disponible, et le processus de mise à jour a commencé.

`STAGING`

L'agent a commencé le téléchargement de la mise à jour. Si l'agent ne peut pas télécharger la mise à jour, ou si le contenu de la mise à jour est incorrect ou endommagé, l'agent envoie une notification de l'échec et la mise à jour passe en état `FAILED`.

`STAGED`

Le téléchargement est terminé et le contenu de l'agent a été vérifié.

`UPDATING`

Le service `ecs-init` est redémarré et il récupère la nouvelle version de l'agent. Si, pour une raison quelconque, l'agent est incapable de redémarrer, la mise à jour passe à l'état `FAILED`. Dans le cas contraire, l'agent indique à Amazon ECS que la mise à jour est terminée.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Pour mettre à jour l'agent de conteneur Amazon ECS sur un Amazon ECS-optimized AMI dans la console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters et sélectionnez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet.
5. Sous Instances de conteneur, sélectionnez les instances à mettre à jour, puis choisissez Actions et Mettre à jour l'agent.

Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-Optimized AMI)

L'agent de conteneur Amazon ECS doit parfois être mis à jour afin d'obtenir les correctifs de bogues et les nouvelles fonctions. La mise à jour de l'agent de conteneur Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution sur l'instance de conteneur.

> **Note**
>
> Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

1. Connectez-vous à votre instance de conteneur via SSH.
2. Vérifiez si votre agent utilise la variable d'environnement `ECS_DATADIR` pour enregistrer son état.

   ```
   ubuntu:~$ docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Sortie :

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Si la commande précédente ne renvoie pas la variable d'environnement `ECS_DATADIR`, vous devez arrêter toutes les tâches en cours d'exécution sur cette instance de conteneur avant de mettre à jour votre agent. Les agents récents avec la variable d'environnement `ECS_DATADIR` enregistrent leur état et sont possibles à mettre à jour sans problèmes tandis que les tâches sont en cours d'exécution.

3. Arrêtez l'agent de conteneur Amazon ECS.

   ```
   ubuntu:~$ docker stop ecs-agent
   ```

4. Supprimez le conteneur de l'agent.

```
ubuntu:~$ docker rm ecs-agent
```

5. Vérifiez que le répertoire `/etc/ecs` et le fichier de configuration de l'agent de conteneur Amazon ECS existent à l'emplacement `/etc/ecs/ecs.config`.

```
ubuntu:~$ sudo mkdir -p /etc/ecs && sudo touch /etc/ecs/ecs.config
```

6. Modifiez le fichier `/etc/ecs/ecs.config` et vérifiez qu'il contient au moins les déclarations de variable suivantes. Pour que votre instance de conteneur ne s'inscrive pas auprès du cluster par défaut, spécifiez le nom de votre cluster comme valeur pour `ECS_CLUSTER`.

```
ECS_DATADIR=/data
ECS_ENABLE_TASK_IAM_ROLE=true
ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true
ECS_LOGFILE=/log/ecs-agent.log
ECS_AVAILABLE_LOGGING_DRIVERS=["json-file","awslogs"]
ECS_LOGLEVEL=info
ECS_CLUSTER=default
```

Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous pouvez éventuellement stocker vos variables d'environnement d'agent dans Amazon S3. Elles pourront être téléchargées dans vos instances de conteneur lors du lancement à l'aide des données utilisateur Amazon EC2. Nous recommandons cette approche pour les informations sensibles telles que les informations d'authentification pour les référentiels privés. Pour plus d'informations, consultez Stockage de la configuration d'instance de conteneur Amazon ECS dans Amazon S3 et En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

7. Extrayez la dernière image de l'agent de conteneur Amazon ECS depuis Amazon Elastic Container Registry Public.

```
ubuntu:~$ docker pull public.ecr.aws/ecs/amazon-ecs-agent:latest
```

Sortie :

```
Pulling repository amazon/amazon-ecs-agent
a5a56a5e13dc: Download complete
511136ea3c5a: Download complete
9950b5d678a1: Download complete
c48ddcf21b63: Download complete
Status: Image is up to date for amazon/amazon-ecs-agent:latest
```

8. Exécutez l'agent de conteneur Amazon ECS le plus récent sur votre instance de conteneur.

> **Note**
>
> Utilisez les stratégies de redémarrage de Docker ou un gestionnaire de processus (tel que upstart ou systemd) pour traiter l'agent de conteneur comme un service ou un démon, et vous assurer qu'il est redémarré après avoir été arrêté. L'Amazonie ECS-optimized AMI utilise le `ecs-init` RPM à cette fin, et vous pouvez consulter le code source pour ce RPM sur GitHub.

L'exemple suivant de commande d'exécution d'agent est divisé en lignes distinctes pour montrer chaque option. Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Important**
>
> Les systèmes d'exploitation avec SELinux activé exigent l'option `--privileged` dans votre commande docker run. En outre, pour SELinux-enabled instances de conteneur, nous vous recommandons d'ajouter le `:Z` option pour `/log` et `/data` supports de volume. Cependant, les montages hôtes de ces volumes doivent exister avant d'exécuter la commande. Dans le cas contraire, vous recevrez une erreur `no such file or directory`. Procédez comme suit si vous rencontrez des difficultés pour exécuter l'agent Amazon ECS sur un SELinux-enabled instance de conteneur :
>
> - Créez les points de montage des volumes hôtes sur votre instance de conteneur.
>
>   ```
>   ubuntu:~$ sudo mkdir -p /var/log/ecs /var/lib/ecs/data
>   ```
>
> - Ajoutez le `--privileged` option pour docker run commande qui suit.

- Ajoutez le `:Z` option pour `/log` et `/data` supports de volume du conteneur (par exemple, `--volume=/var/log/ecs/:/log:Z`) au docker run commande qui suit.

```
ubuntu:~$ sudo docker run --name ecs-agent \
--detach=true \
--restart=on-failure:10 \
--volume=/var/run:/var/run \
--volume=/var/log/ecs/:/log \
--volume=/var/lib/ecs/data:/data \
--volume=/etc/ecs:/etc/ecs \
--volume=/etc/ecs:/etc/ecs/pki \
--net=host \
--env-file=/etc/ecs/ecs.config \
amazon/amazon-ecs-agent:latest
```

> **Note**
>
> Si vous recevez un message `Error response from daemon: Cannot start container`, vous pouvez supprimer le conteneur ayant échoué à l'aide de la commande sudo docker rm ecs-agent, puis réessayer d'exécuter l'agent.

Amazon ECS-optimized AMI Windows

L'Amazonie ECS-optimized Les AMI sont préconfigurées avec les composants nécessaires pour exécuter les charges de travail Amazon ECS. Bien que vous puissiez créer votre propre instance de conteneur AMI répondant aux spécifications de base requises pour exécuter vos charges de travail conteneurisées sur Amazon ECS, Amazon ECS-optimized Les AMI sont préconfigurées et testées sur Amazon ECS par AWS ingénieurs. C'est la façon la plus simple de démarrer et d'exécuter rapidement vos conteneurs sur AWS .

L'Amazonie ECS-optimized Les métadonnées de l'AMI, y compris le nom de l'AMI, la version de l'agent de conteneur Amazon ECS et la version d'exécution d'Amazon ECS qui inclut la version Docker, peuvent être récupérées par programmation pour chaque variante. Pour plus d’informations, consultez the section called “Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows”.

> **Important**
>
> Tous ECS-optimized Les variantes AMI produites après août 2022 migreront de Docker EE (Mirantis) vers Docker CE (projet Moby).
> Pour s'assurer que les clients disposent des dernières mises à jour de sécurité par défaut, Amazon ECS gère au moins les trois dernières versions de Windows Amazon ECS-optimized AMI. Après la sortie de la nouvelle version de Windows, Amazon ECS-optimized AMI, Amazon ECS fait de Windows Amazon ECS-optimized Les AMI qui sont plus anciennes sont privées. S'il existe une AMI privée à laquelle vous devez accéder, soumettez un ticket auprès du support Cloud.

Amazon ECS-optimized Variantes AMI

Les variantes de Windows Server suivantes d'Amazon ECS-optimized Des AMI sont disponibles pour vos instances Amazon EC2.

> **Important**
>
> Tous ECS-optimized Les variantes AMI produites après le mois d'août seront migrées de Docker EE (Mirantis) vers Docker CE (projet Moby).

- Amazon ECS-optimized AMI complète pour Windows Server 2025
- Amazon ECS-optimized AMI principale pour Windows Server 2025
- Amazon ECS-optimized AMI complète pour Windows Server 2022
- Amazon ECS-optimized AMI principale pour Windows Server 2022
- Amazon ECS-optimized AMI complète pour Windows Server 2019
- Amazon ECS-optimized AMI principale pour Windows Server 2019
- Amazon ECS-optimized AMI complète pour Windows Server 2016

> **Important**
>
> Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, les AMI Windows Server 2016 Full ne recevront pas de correctifs

de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Le 9 août 2022, l'Amazonie ECS-optimized Windows Server 20H2 Core AMI a atteint sa date de fin de support. Aucune nouvelle version de cette AMI ne sera publiée. Pour plus d'informations, consultez les informations sur la version de Windows Server.

Windows Server 2025, Windows Server 2022, Windows Server 2019 et Windows Server 2016 sont Long-Term Publications de Servicing Channel (LTSC). Windows Server 20H2 est un Semi-Annual Communiqué de la chaîne (SAC). Pour plus d'informations, consultez les informations sur la version de Windows Server.

Considérations

Voici quelques éléments que vous devez savoir sur les conteneurs Windows Amazon EC2 et Amazon ECS.

- Les conteneurs Windows ne peuvent pas s'exécuter sur des instances de conteneur Linux, et inversement. Afin de garantir un meilleur placement de la tâche pour les tâches Windows et Linux, gardez les instances de conteneur Windows et Linux dans des clusters distincts et placez uniquement les tâches Windows sur des clusters Windows. Pour vous assurer que les définitions de tâche Windows sont uniquement placées sur des instances Windows, vous pouvez définir la contrainte de placement suivante : `memberOf(ecs.os-type=='windows')`.
- Les conteneurs Windows sont pris en charge pour les tâches qui utilisent EC2 et Fargate.
- Les conteneurs Windows et les instances de conteneur ne peuvent pas prendre en charge tous les paramètres de définition de tâche disponibles pour les conteneurs Linux et les instances de conteneur. Certains paramètres ne sont pas du tout pris en charge, tandis que d'autres se comportent différemment sous Windows et Linux. Pour plus d'informations, consultez Différences entre les définitions de tâche Amazon ECS pour les instances EC2 exécutant Windows.
- Pour les rôles IAM pour la fonction des tâches, vous devez configurer vos instances de conteneur Windows pour autoriser la fonction lors du lancement. Vos conteneurs doivent fonctionner

certains PowerShell code lorsqu'ils utilisent la fonctionnalité. Pour plus d'informations, consultez Configuration supplémentaire d'instance Windows Amazon EC2.

- Les rôles IAM pour la fonction des tâches utilisent un proxy d'informations d'identification pour fournir des informations d'identification aux conteneurs. Ce proxy d'informations d'identification occupe le port 80 sur l'instance de conteneur, donc si vous utilisez des rôles IAM pour les tâches, le port 80 n'est pas disponible pour les tâches. Pour les conteneurs de service web, vous pouvez utiliser un Application Load Balancer et un mappage de port dynamique pour fournir des connexions port 80 HTTP standard à vos conteneurs. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.
- Les images Docker de Windows Server sont volumineuses (9 Gio). Ainsi, vos instances de conteneur Windows nécessitent plus d'espace de stockage que les instances de conteneur Linux.
- Pour exécuter un conteneur Windows sur un Windows Server, la version du système d'exploitation de l'image de base du conteneur doit correspondre à celle de l'hôte. Pour plus d'informations, consultez Compatibilité avec la version du conteneur Windows sur le site web de documentation Microsoft. Si votre cluster exécute plusieurs versions de Windows, vous pouvez vous assurer qu'une tâche est placée sur une instance EC2 exécutée sur la même version en utilisant la contrainte de placement : `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour plus d'informations, consultez the section called "Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows".

Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows

L'ID AMI, le nom de l'image, le système d'exploitation, la version de l'agent de conteneur et la version d'exécution pour chaque variante d'Amazon ECS-optimized Les AMI peuvent être récupérées par programmation en interrogeant l'API Systems Manager Parameter Store. Pour plus d'informations sur l'API Systems Manager Parameter Store, voir GetParameters et GetParametersByPath.

> **Note**
>
> Votre utilisateur administratif doit disposer des autorisations IAM suivantes pour récupérer l'Amazon ECS-optimized Métadonnées AMI. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`.
>
> - sms :GetParameters
> - sms :GetParameter
> - sms :GetParametersByPath

Format de paramètre Systems Manager Parameter Store

> **Note**
>
> Les paramètres d'API Systems Manager Parameter Store suivants sont obsolètes et ne doivent pas être utilisés pour récupérer les dernières AMI Windows :
>
> - `/aws/service/ecs/optimized-ami/windows_server/2016/english/full/recommended/image_id`
> - `/aws/service/ecs/optimized-ami/windows_server/2019/english/full/recommended/image_id`

Voici le format du nom des paramètres pour chaque Amazon ECS-optimized Variante AMI.

- Métadonnées de l'AMI Windows Server 2025 Full :

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2025 Core :

```
/aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2022 Full :

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2022 Core :

```
/aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI Windows Server 2019 Full :

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

- Métadonnées de l'AMI principale de Windows Server 2019 :

```
/aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

- Métadonnées de l'AMI complète de Windows Server 2016 :

```
/aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Le format de nom de paramètre suivant extrait les métadonnées de la dernière AMI de base complète Windows Server 2019.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-
English-Full-ECS_Optimized
```

Les informations suivantes présentent un exemple de l'objet JSON renvoyé pour la valeur du paramètre.

```
{
    "Parameters": [
        {
            "Name": "/aws/service/ami-windows-latest/Windows_Server-2019-English-Full-
ECS_Optimized",
            "Type": "String",
            "Value": "{\"image_name\":\"Windows_Server-2019-English-Full-
ECS_Optimized-2023.06.13\",\"image_id\":\"ami-0debc1fb48e4aee16\",\"ecs_runtime_version
\":\"Docker (CE) version 20.10.21\",\"ecs_agent_version\":\"1.72.0\"}",
            "Version": 58,
            "LastModifiedDate": "2023-06-22T19:37:37.841000-04:00",
            "ARN": "arn:aws:ssm:us-east-1::parameter/aws/service/ami-windows-latest/
Windows_Server-2019-English-Full-ECS_Optimized",
            "DataType": "text"
        }
    ],
    "InvalidParameters": []
}
```

Chacun des champs de la sortie précédente peut être interrogé en tant que sous-paramètres. Construisez le chemin d'accès d'un sous-paramètre en ajoutant le nom du sous-paramètre au chemin d'accès de l'AMI sélectionnée. Les sous-paramètres suivants sont disponibles :

- `schema_version`
- `image_id`
- `image_name`
- `os`

- `ecs_agent_version`
- `ecs_runtime_version`

Exemples

Les exemples suivants montrent comment vous pouvez récupérer les métadonnées de chaque Amazon. ECS-optimized Variante AMI.

Récupération des métadonnées de la dernière version stable d'Amazon ECS-optimized AMI

Vous pouvez récupérer la dernière version stable d'Amazon ECS-optimized AMI à l'aide du AWS CLI avec les éléments suivants AWS CLI commandes.

- Pour l'Amazon ECS-optimized AMI complète pour Windows Server 2025 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI principale pour Windows Server 2025 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI complète pour Windows Server 2022 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI principale pour Windows Server 2022 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI complète pour Windows Server 2019 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized --region us-east-1
```

- Pour l'Amazon ECS-optimized AMI principale pour Windows Server 2019 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized --region us-east-1
```

- Pour l'Amazonie ECS-optimized AMI complète pour Windows Server 2016 :

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized --region us-east-1
```

Utilisation de la dernière version recommandée d'Amazon ECS-optimized AMI dans un CloudFormation modèle

Vous pouvez consulter la dernière version recommandée d'Amazon ECS-optimized AMI dans un CloudFormation modèle en faisant référence au nom du magasin de paramètres de Systems Manager.

```
Parameters:
  LatestECSOptimizedAMI:
    Description: AMI ID
    Type: AWS::SSM::Parameter::Value<AWS::EC2::Image::Id>
    Default: /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized/image_id
```

Amazon ECS-optimized Versions de Windows AMI

Afficher les versions actuelles et précédentes d'Amazon ECS-optimized Les AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS, Docker, et du `ecs-init` colis.

L'Amazonie ECS-optimized Les métadonnées AMI, y compris l'ID AMI, pour chaque variante peuvent être récupérées par programmation. Pour plus d'informations, consultez the section called "Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows".

Les onglets suivants affichent une liste de Windows Amazon ECS-optimized Versions AMI. Pour plus de détails sur le référencement du paramètre Systems Manager Parameter Store dans un CloudFormation modèle, voir Utilisation de la dernière version recommandée d'Amazon ECS-optimized AMI dans un CloudFormation modèle.

Important

Pour garantir que les clients disposent des dernières mises à jour de sécurité par défaut, Amazon ECS gère au moins les trois dernières versions de Windows Amazon ECS-optimized AMI. Après la sortie de la nouvelle version de Windows, Amazon ECS-optimized AMI, Amazon ECS fait de Windows Amazon ECS-optimized Les AMI qui sont plus anciennes sont privées. S'il existe une AMI privée à laquelle vous devez accéder, soumettez un ticket auprès du support Cloud.
Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, les AMI Windows Server 2016 Full ne recevront pas de correctifs de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Note

La journalisation du plug-in gMSA est passée d'une journalisation basée sur des fichiers (`C:\ProgramData\Amazon\gmsa`) vers Windows Event logging lors de la version d'août 2025 de l'AMI. Le script public log collector collectera tous les logs gMSA. Pour plus d'informations, consultez Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS.

Windows Server 2025 Full AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2025 Full AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI complète pour Windows Server 2025	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2025-English-Full-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Full-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Full-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Full-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI complète pour Windows Server 2025.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Full-ECS_Optimized
```

Windows Server 2025 Core AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2025 Core AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI principale pour Windows Server 2025	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2025-English-Core-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Core-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Core-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2025-English-Core-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI principale Windows Server 2025.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2025-English-Core-ECS_Optimized
```

Windows Server 2022 Full AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2022 Full AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI complète pour Windows Server 2022	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Full-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Full-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI complète pour Windows Server 2022.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Full-ECS_Optimized
```

Windows Server 2022 Core AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2022 Core AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI principale pour Windows Server 2022	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2022-English-Core-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2022-English-Core-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI complète pour Windows Server 2022.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2022-English-Core-ECS_Optimized
```

Windows Server 2019 Full AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2019 Full AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI complète pour Windows Server 2019	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Full-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Full-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI complète pour Windows Server 2019.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Full-ECS_Optimized
```

Windows Server 2019 Core AMI versions

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2019 Core AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI principale pour Windows Server 2019	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2019-English-Core-ECS_Optimized-2026.10.16	1.108.0	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2026.09.12	1.106.2	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2026.08.15	1.106.1	25.0.16 (Docker CE)	Public
Windows_Server-2019-English-Core-ECS_Optimized-2026.07.18	1.105.1	25.0.16 (Docker CE)	Public

Utilisez ce qui suit AWS CLI commande pour récupérer l'Amazon actuel ECS-optimized AMI complète pour Windows Server 2019.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2019-English-Core-ECS_Optimized
```

Windows Server 2016 Full AMI versions

> **Important**
>
> Windows Server 2016 ne prend pas en charge la dernière version de Docker, par exemple 25.x.x. Par conséquent, les AMI Windows Server 2016 Full ne recevront pas de correctifs

de sécurité ou de bogues pour le moteur d'exécution Docker. Nous vous recommandons de passer à l'une des plateformes Windows suivantes :

- Windows Server 2022 Full
- Windows Server 2022 Core
- Windows Server 2019 Full
- Windows Server 2019 Core

Le tableau suivant répertorie les versions actuelles et précédentes d'Amazon. ECS-optimized Windows Server 2016 Full AMI et leurs versions correspondantes de l'agent de conteneur Amazon ECS et de Docker.

Amazon ECS-optimized AMI complète pour Windows Server 2016	Version d'agent de conteneur Amazon ECS	Version de Docker	Visibilité
Windows_Server-2016-English-Full-ECS_Optimized-2026.10.16	`1.108.0`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2026.09.12	`1.106.2`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2026.08.15	`1.106.1`	`20.10.23 (Docker CE)`	Public
Windows_Server-2016-English-Full-ECS_Optimized-2026.07.18	`1.105.1`	`20.10.23 (Docker CE)`	Public

Utilisez ce qui suit AWS CLI Amazon ECS-optimized AMI complète pour Windows Server 2016.

```
aws ssm get-parameters --names /aws/service/ami-windows-latest/Windows_Server-2016-English-Full-ECS_Optimized
```

Créez votre propre Amazon ECS-optimized AMI pour Windows

Utilisez EC2 Image Builder pour créer votre propre Amazon personnalisé ECS-optimized AMI pour Windows. Cela facilite l'utilisation d'une AMI Windows avec votre propre licence sur Amazon ECS. Amazon ECS propose un composant Image Builder géré qui fournit la configuration système nécessaire pour exécuter des instances Windows afin d'héberger vos conteneurs. Chaque composant géré par Amazon ECS comprend un agent de conteneur spécifique et une version Docker. Vous pouvez personnaliser votre image pour utiliser le dernier composant géré Amazon ECS, ou si un agent de conteneur plus ancien ou une version Docker plus ancienne est nécessaire, vous pouvez spécifier un autre composant.

Pour une démonstration complète de l'utilisation d'EC2 Image Builder, veuillez consulter Getting started with EC2 Image Builder dans le Guide de l'utilisateur EC2 Image Builder.

Lorsque vous créez votre propre Amazon ECS-optimized Windows AMI à l'aide d'EC2 Image Builder, vous créez une recette d'image. Votre recette d'image doit répondre aux critères suivants :

- L'image source doit être basée sur Windows Server 2019 Core, Windows Server 2019 Full, Windows Server 2022 Core ou Windows Server 2022 Full. Tout autre système d'exploitation Windows n'est pas pris en charge et peut ne pas être compatible avec le composant.
- Lorsque vous spécifiez les composants de création, le composant `ecs-optimized-ami-windows` est requis. Le `update-windows` Ce composant, que nous recommandons, garantit que l'image contient les dernières mises à jour de sécurité.

  Pour spécifier une version de composant différente, développez le menu Versioning options (Options de gestion des versions) et spécifiez la version du composant à utiliser. Pour plus d'informations, consultez Répertorier les `ami-windows optimisé pour ECS` versions des composants.

Répertorier les **`ami-windows optimisé pour ECS`** versions des composants

Lors de la création d'une recette EC2 Image Builder et de la spécification du composant `ecs-optimized-ami-windows`, vous pouvez utiliser l'option par défaut ou spécifier une version de

composant spécifique. Pour déterminer les versions de composants disponibles, ainsi que l'agent de conteneur Amazon ECS et les versions Docker contenues dans le composant, vous pouvez utiliser la Console de gestion AWS.

Pour répertorier les produits disponibles **`ami-windows optimisé pour ECS`** versions des composants

1. Ouvrez la console EC2 Image Builder à l'adresse https://console.aws.amazon.com/imagebuilder/.
2. Dans la barre de navigation, sélectionnez la région dans laquelle votre image est créée.
3. Dans le panneau de navigation, sous le menu Saved configurations (Configurations enregistrées), choisissez Components (Composants).
4. Sur le Composantes page, dans la barre de recherche, tapez `ecs-optimized-ami-windows` et déroulez le menu de qualification et sélectionnez Démarrage rapide (Amazon-managed).
5. Utilisation de la colonne Description pour déterminer la version du composant avec l'agent de conteneur Amazon ECS et la version Docker qu'exige votre image.

Gestion des instances de conteneurs Windows Amazon ECS

Lorsque vous utilisez des instances EC2 pour vos charges de travail Amazon ECS, vous êtes responsable de la maintenance des instances.

Les mises à jour de l'agent ne s'appliquent pas aux instances de conteneur Windows. Nous vous recommandons de lancer de nouvelles instances de conteneur pour mettre à jour le version de l'agent dans vos clusters Windows.

Procédures de gestion

- Lancement d'une instance de conteneur Amazon ECS Windows
- Amorçage des instances de conteneurs Windows Amazon ECS pour transmettre des données
- Utilisation d’un proxy HTTP pour les instances de conteneur Windows Amazon ECS
- Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Lancement d'une instance de conteneur Amazon ECS Windows

Vos instances de conteneur Amazon ECS sont créées à l'aide de la console Amazon EC2. Avant de commencer, assurez-vous d'avoir terminé les étapes de Configurer l'utilisation d'Amazon ECS.

Pour plus d'informations sur l'assistant de lancement, consultez la section Lancement d'une instance à l'aide du nouvel assistant de lancement d'instance dans le Guide de l'utilisateur Amazon EC2.

Vous pouvez utiliser le nouvel assistant Amazon EC2 pour lancer une instance. Vous pouvez utiliser la liste suivante pour les paramètres et laisser les paramètres non répertoriés comme paramètres par défaut. Les instructions suivantes vous guident dans chaque groupe de paramètres.

Procédure

Avant de commencer, complétez les étapes détaillées dans Configurer l'utilisation d'Amazon ECS.

1. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
2. Dans la barre de navigation en haut de l'écran, l'actuel AWS La région est affichée (par exemple, USA Est (Ohio)). Sélectionnez une région dans laquelle lancer l'instance. Ce choix est important car certaines ressources Amazon EC2 peuvent être partagées entre des régions, contrairement à d'autres ressources.
3. Sur le tableau de bord de la console Amazon EC2, sélectionnez Launch instance (Lancer une instance).

Noms et identifications

Le nom de l'instance est une identification, où la clé est Name (Nom), et la valeur est le nom que vous spécifiez. Vous pouvez étiqueter l'instance, les volumes et les Elastic Graphics. Pour les instances Spot, vous pouvez baliser uniquement la demande d'instance Spot.

La spécification d'un nom d'instance et d'identifications supplémentaires est facultative.

- Pour Name (Nom), saisissez un nom descriptif pour l'instance. Si vous ne spécifiez pas de nom, l'instance peut être identifiée par son ID, qui est automatiquement généré lorsque vous lancez l'instance.
- Pour ajouter des identifications supplémentaires, sélectionnez Add additional tags (Ajouter des identifications supplémentaires). Choisissez Add tag (Ajouter une identification), saisissez une clé et une valeur, puis sélectionnez le type de ressource à étiqueter. Choisissez Add tag (Ajouter une identification) pour chaque étiquette supplémentaire.

Images d’applications et de systèmes d’exploitation (Amazon Machine Image)

Une Amazon Machine Image (AMI) contient les informations requises pour créer une instance. Par exemple, une AMI peut contenir le logiciel nécessaire pour fonctionner en tant que serveur web, comme Apache et votre site web.

Pour la dernière version d'Amazon ECS-optimized Les AMI et leurs valeurs, voir Fenêtres Amazon ECS-optimized AMI.

Utilisez le Rechercher barre pour trouver un Amazon approprié ECS-optimized AMI publié par AWS.

1. Selon vos besoins, saisissez l'une des AMI suivantes dans la barre de recherche et appuyez sur la touche Enter (Entrée).
   - Windows_Server-2022-English-Full-ECS_Optimisé
   - Windows_Server-2022-English-Core-ECS_Optimisé
   - Windows_Server-2019-English-Full-ECS_Optimisé
   - Windows_Server-2019-English-Core-ECS_Optimisé
   - Windows_Server-2016-English-Full-ECS_Optimisé
2. Sur la page Sélection d'une Amazon Machine Image (AMI) sélectionnez l'onglet AMI de communauté.
3. Dans la liste qui apparaît, choisissez Microsoft-verified AMI avec la date de publication la plus récente et cliquez sur Sélectionnez.

Type d’instance

Le type d’instance définit la configuration matérielle et la taille de l’instance. Les types d’instance plus importants disposent de plus d’UC et de mémoire. Pour plus d’informations, consultez Types d’instance.

- Pour Instance type (Type d’instance), sélectionnez le type de l’instance.

  Le type d'instance que vous sélectionnez détermine les ressources disponibles pour les tâches à exécuter.

Paire de clés (connexion)

Pour Key pair name (Nom de la paire de clés), choisissez une paire de clés existante ou choisissez Create new key pair (Créer une nouvelle paire de clés) pour en créer une nouvelle.

> **Important**
>
> Si vous sélectionnez l'option Proceed without key pair (Not recommended) ((Continuer sans paire de clé) (Non recommandé)), vous ne pourrez pas vous connecter à l'instance à moins de choisir une AMI configurée de façon à autoriser les utilisateurs à se connecter d'une autre façon.

Paramètres réseau

Configurez les paramètres réseau, le cas échéant.

- Networking platform (Plateforme de mise en réseau) : choisissez Virtual Private Cloud (VPC) (Cloud privé virtuel [VPC]), puis spécifiez le sous-réseau dans la section Network interfaces (Interfaces réseau).
- VPC : sélectionnez un VPC existant dans lequel créer le groupe de sécurité.
- Sous-réseau : vous pouvez lancer une instance dans un sous-réseau associé à une zone de disponibilité, une zone locale, une zone Wavelength ou un Outpost.

  Pour lancer l'instance dans une zone de disponibilité, sélectionnez le sous-réseau dans lequel lancer votre instance. Pour créer un sous-réseau, choisissez Créer un nouveau sous-réseau afin d'accéder à la console Amazon VPC. Une fois que vous avez terminé, revenez dans l'assistant de lancement d'instance et choisissez l'icône Refresh (Actualiser) afin de charger votre sous-réseau dans la liste.

  Pour lancer l'instance dans une zone locale, sélectionnez un sous-réseau que vous avez créé dans la zone locale.

  Pour lancer une instance dans un Outpost, sélectionnez un sous-réseau dans un VPC que vous avez associé à l'Outpost.
- Auto-assign IP publique: Si votre instance doit être accessible depuis Internet, vérifiez que Auto-assign IP publique le champ est défini sur Activer. Si ce n'est pas le cas, définissez ce champ sur Disable (Désactiver).

> **Note**
>
> Les instances de conteneur ont besoin de communiquer avec le point de terminaison de service Amazon ECS. Cela peut être via un point de terminaison d'un VPC d'interface ou via vos instances de conteneur ayant des adresses IP publiques.
> Pour plus d'informations sur les points de terminaison d'un VPC d'interface, consultez Points de terminaison VPC de l'interface Amazon ECS (AWS PrivateLink).
> Si vous n'avez pas de point de terminaison d'un VPC d'interface configuré et que vos instances de conteneur n'ont pas d'adresses IP publiques, elles doivent utiliser la traduction d'adresses réseau (NAT) pour fournir cet accès. Pour de plus amples informations, veuillez consulter Passerelles NAT dans le Guide de l'utilisateur Amazon VPC et Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux dans ce guide.

- Firewall (security groups) (Pare-feu (groupes de sécurité)) : utilisez un groupe de sécurité afin de définir les règles de pare-feu de votre instance de conteneur. Ces règles déterminent le trafic réseau entrant acheminé vers votre instance de conteneur. Le reste du trafic est ignoré.
  - Pour sélectionner un groupe de sécurité existant, choisissez Select existing security group (Sélectionner un groupe de sécurité existant), puis sélectionnez le groupe de sécurité que vous avez créé dans Configurer l'utilisation d'Amazon ECS.

Configurer le stockage

L'AMI sélectionnée inclut un ou plusieurs volumes de stockage, notamment le volume racine. Vous pouvez spécifier d'autres volumes à attacher à l'instance.

Vous pouvez utiliser la vue Simple.

- Storage type (Type de stockage) : configurez le stockage pour votre instance de conteneur.

  Si vous utilisez Amazon ECS-optimized Amazon Linux AMI, deux volumes sont configurés pour votre instance. Le volume Racine est réservé au système d'exploitation et le second volume Amazon EBS (attaché à `/dev/xvdcz`) est réservé à Docker.

  Vous pouvez augmenter ou diminuer la taille des volumes pour votre instance afin de répondre aux besoins de votre application.

Détails avancés

Développez la section Détails avancés pour afficher les champs et spécifier des paramètres supplémentaires pour l'instance.

- Purchasing option (Option d'achat) : sélectionnez Request Spot Instances (Demander des instances Spot) pour demander une instance Spot. Vous devez également définir les autres champs associés aux instances Spot. Pour plus d'informations, consultez Demandes d'instances Spot.

  > **Note**
  >
  > Si vous utilisez les instances Spot et qu'un message `Not available` s'affiche, vous devrez peut-être choisir un autre type d'instance.

  .

- IAM instance profile (Profil d'instance IAM) : sélectionnez votre rôle IAM d'instance de conteneur. Celui-ci est généralement nommé `ecsInstanceRole`.

  > **Important**
  >
  > Si vous ne pas lancez votre instance de conteneur avec les autorisations IAM appropriées, votre agent Amazon ECS ne peut pas se connecter à votre cluster. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

- (Facultatif) User data (Données utilisateur) : configurez votre instance de conteneur Amazon ECS avec les données utilisateur, telles que les variables d'environnement d'agent depuis Configuration de l'agent de conteneur Amazon ECS. Les scripts de données utilisateur Amazon EC2 sont exécutés une seule fois, au premier lancement de l'instance. Les exemples suivants présentent des utilisations courantes des données utilisateur :

  - Par défaut, votre instance de conteneur est lancée dans votre cluster par défaut. Pour un lancement dans un cluster autre que celui défini par défaut, choisissez la liste Détails avancés. Collez ensuite le script suivant dans Données de l'utilisateur champ, remplacement *`your_cluster_name`* avec le nom de votre cluster.

    `EnableTaskIAMRole` active la fonction Rôles IAM pour les tâches.

En outre, les options suivantes sont disponibles lorsque vous utilisez le mode réseau `awsvpc`.

- `EnableTaskENI` : cet indicateur active les réseaux des tâches et est requis lorsque vous utilisez le mode réseau `awsvpc`.
- `AwsvpcBlockIMDS` : cet indicateur facultatif bloque l'accès IMDS pour les conteneurs de tâches qui s'exécutent dans en mode réseau `awsvpc`.
- `AwsvpcAdditionalLocalRoutes` : cet indicateur facultatif vous permet d'avoir des acheminements supplémentaires dans l'espace de noms de tâche.

  Remplacer `ip-address` avec l'adresse IP pour les itinéraires supplémentaires, par exemple 172.31.42.23/32.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster your_cluster_name -EnableTaskIAMRole -EnableTaskENI -
AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Amorçage des instances de conteneurs Windows Amazon ECS pour transmettre des données

Lorsque vous lancez une instance Amazon EC2, vous pouvez transmettre les données utilisateur à cette instance. Ces données peuvent être utilisées afin d'effectuer des tâches de configuration automatisées courantes, voire d'exécuter des scripts lors du démarrage de l'instance. Pour Amazon ECS, les données utilisateur sont le plus souvent utilisées pour transmettre les informations de configuration au démon Docker et à l'agent de conteneur Amazon ECS.

Vous pouvez transmettre plusieurs types de données utilisateur à Amazon EC2, notamment des boothooks de cloud, des scripts shell et des directives `cloud-init`. Pour plus d'informations sur ces types de formats et d'autres, consultez le Cloud-Init documentation.

Vous pouvez transmettre ces données utilisateur lors de l'utilisation de l'Amazon EC2 Launch Wizard. Pour plus d’informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Données utilisateur Windows par défaut

Cet exemple de script de données utilisateur montre les données utilisateur par défaut que vos instances de conteneur Windows reçoivent si vous utilisez la console. Le script suivant effectue les opérations suivantes :

- Définit le nom du cluster selon le nom que vous avez saisi.
- Définit les rôles IAM pour les tâches.
- Définit `json-file` et `awslogs` comme pilotes de journalisation disponibles.

En outre, les options suivantes sont disponibles lorsque vous utilisez le mode réseau `awsvpc`.

- `EnableTaskENI` : cet indicateur active les réseaux des tâches et est requis lorsque vous utilisez le mode réseau `awsvpc`.
- `AwsvpcBlockIMDS` : cet indicateur facultatif bloque l'accès IMDS pour les conteneurs de tâches qui s'exécutent dans en mode réseau `awsvpc`.
- `AwsvpcAdditionalLocalRoutes` : cet indicateur facultatif vous permet d'avoir des acheminements supplémentaires.

  Remplacer `ip-address` avec l'adresse IP pour les itinéraires supplémentaires, par exemple 172.31.42.23/32.

Vous pouvez utiliser ce script pour vos propres instances de conteneurs (à condition qu'elles soient lancées depuis Amazon). ECS-optimized Windows Server (AMI).

Remplacez la ligne `-Cluster` *`cluster-name`* pour spécifier votre propre nom de cluster.

```
<powershell>
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]' -EnableTaskENI -AwsvpcBlockIMDS -AwsvpcAdditionalLocalRoutes
'["ip-address"]'
</powershell>
```

Pour les tâches Windows configurées pour utiliser le pilote de journalisation `awslogs`, vous devez également définir la variable d'environnement `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE` sur votre instance de conteneur. Utilisez la syntaxe suivante.

Remplacez la ligne `-Cluster` *`cluster-name`* pour spécifier votre propre nom de cluster.

```
<powershell>
[Environment]::SetEnvironmentVariable("ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE",
 $TRUE, "Machine")
```

```
Initialize-ECSAgent -Cluster cluster-name -EnableTaskIAMRole -LoggingDrivers '["json-
file","awslogs"]'
</powershell>
```

Données utilisateur d'installation d'un agent Windows

Cet exemple de script de données utilisateur installe l'agent de conteneur Amazon ECS sur une instance lancée avec un Windows_Server-2016-English-Full-Containers AMI. Il a été adapté à partir des instructions d'installation de l'agent sur le Agent de conteneurs Amazon ECS GitHub référentiel Page README.

> **Note**
>
> Ce script est partagé à titre d'exemple. Il est beaucoup plus facile de démarrer avec les conteneurs Windows en utilisant Amazon ECS-optimized AMI Windows Server. Pour plus d'informations, consultez Création d'un cluster Amazon ECS pour les charges de travail Fargate.

Pour plus d'informations sur la façon d'installer l'agent Amazon ECS sur Windows Server 2022 Full, consultez Numéro 3753 sur GitHub.

Vous pouvez utiliser ce script pour vos propres instances de conteneurs (à condition qu'elles soient lancées avec une version de Windows_Server-2016-English-Full-Containers AMI). Veillez à modifier la ligne *windows* dans le fichier de configuration en spécifiant votre propre nom de cluster (si vous n'utilisez pas un cluster nommé windows).

```
<powershell>
# Set up directories the agent uses
New-Item -Type directory -Path ${env:ProgramFiles}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS -Force
New-Item -Type directory -Path ${env:ProgramData}\Amazon\ECS\data -Force
# Set up configuration
$ecsExeDir = "${env:ProgramFiles}\Amazon\ECS"
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", "windows", "Machine")
[Environment]::SetEnvironmentVariable("ECS_LOGFILE", "${env:ProgramData}\Amazon\ECS\log
\ecs-agent.log", "Machine")
[Environment]::SetEnvironmentVariable("ECS_DATADIR", "${env:ProgramData}\Amazon\ECS
\data", "Machine")
# Download the agent
$agentVersion = "latest"
```

```
$agentZipUri = "https://s3.amazonaws.com/amazon-ecs-agent/ecs-agent-windows-
$agentVersion.zip"
$zipFile = "${env:TEMP}\ecs-agent.zip"
Invoke-RestMethod -OutFile $zipFile -Uri $agentZipUri
# Put the executables in the executable directory.
Expand-Archive -Path $zipFile -DestinationPath $ecsExeDir -Force
Set-Location ${ecsExeDir}
# Set $EnableTaskIAMRoles to $true to enable task IAM roles
# Note that enabling IAM roles will make port 80 unavailable for tasks.
[bool]$EnableTaskIAMRoles = $false
if (${EnableTaskIAMRoles}) {
  $HostSetupScript = Invoke-WebRequest https://raw.githubusercontent.com/aws/amazon-
ecs-agent/master/misc/windows-deploy/hostsetup.ps1
  Invoke-Expression $($HostSetupScript.Content)
}
# Install the agent service
New-Service -Name "AmazonECS" `
        -BinaryPathName "$ecsExeDir\amazon-ecs-agent.exe -windows-service" `
        -DisplayName "Amazon ECS" `
        -Description "Amazon ECS service runs the Amazon ECS agent" `
        -DependsOn Docker `
        -StartupType Manual
sc.exe failure AmazonECS reset=300 actions=restart/5000/restart/30000/restart/60000
sc.exe failureflag AmazonECS 1
Start-Service AmazonECS
</powershell>
```

Utilisation d’un proxy HTTP pour les instances de conteneur Windows Amazon ECS

Vous pouvez configurer vos instances de conteneur Amazon ECS pour qu'elles utilisent un proxy HTTP pour l'agent de conteneur Amazon ECS et le démon Docker. C'est utile si vos instances de conteneur n'ont pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

Pour configurer votre instance de conteneur Amazon ECS Windows afin qu'elle utilise un proxy HTTP, définissez les variables suivantes lors du lancement (avec les données utilisateur Amazon EC2).

```
[Environment]::SetEnvironmentVariable("HTTP_PROXY",
"http://proxy.mydomain:port", "Machine")
```

Définissez HTTP_PROXY sur le nom d'hôte (ou l'adresse IP) et le numéro de port d'un proxy HTTP à utiliser pour que l'agent Amazon ECS puisse se connecter à Internet. Par exemple, vos

instances de conteneur n'ont peut-être pas accès au réseau externe via une passerelle Internet Amazon VPC, une instance ou une passerelle NAT.

```
[Environment]::SetEnvironmentVariable("NO_PROXY",
"169.254.169.254,169.254.170.2,\\.\pipe\docker_engine", "Machine")
```

Définissez `NO_PROXY` sur `169.254.169.254,169.254.170.2,\\.\pipe\docker_engine` pour filtrer les métadonnées d'instance EC2, les rôles IAM pour les tâches et le trafic du démon Docker en provenance du proxy.

Example Script de données utilisateur proxy HTTP pour Windows

L'exemple de données utilisateur suivant PowerShell script configure l'agent de conteneur Amazon ECS et le démon Docker pour qu'ils utilisent un proxy HTTP que vous spécifiez. Vous pouvez également spécifier un cluster auprès duquel l'instance de conteneur s'enregistrera.

Pour utiliser ce script lors du lancement d'une instance de conteneur, suivez les étapes indiquées dans the section called “Lancement d'une instance de conteneur”. Copiez et collez simplement le PowerShell script qui fait suite à Données de l'utilisateur champ (veillez à remplacer les valeurs d'exemple en rouge par vos propres informations de proxy et de cluster).

> **Note**
>
> L'option `-EnableTaskIAMRole` est nécessaire pour activer les rôles IAM pour les tâches. Pour plus d’informations, consultez Configuration supplémentaire d’instance Windows Amazon EC2.

```
<powershell>
Import-Module ECSTools

$proxy = "http://proxy.mydomain:port"
[Environment]::SetEnvironmentVariable("HTTP_PROXY", $proxy, "Machine")
[Environment]::SetEnvironmentVariable("NO_PROXY", "169.254.169.254,169.254.170.2,\\.
\pipe\docker_engine", "Machine")

Restart-Service Docker
Initialize-ECSAgent -Cluster MyCluster -EnableTaskIAMRole
</powershell>
```

Configuration des instances de conteneur Windows Amazon ECS pour recevoir des notifications relatives aux instances Spot

Amazon EC2 résilie, arrête ou met en veille prolongée votre instance Spot lorsque le prix Spot dépasse le prix maximum de votre demande ou lorsque la capacité n'est plus disponible. Amazon EC2 communique un avis d'interruption d'instance Spot, qui donne à l'instance un avertissement deux minutes avant qu'elle soit interrompue. Si le drainage des instances Spot Amazon ECS est activé sur l'instance, ECS reçoit l'avis d'interruption d'instance Spot et bascule l'instance à l'état `DRAINING`.

> **Important**
>
> Amazon ECS surveille les avis d'interruption d'instance Spot qui comportent les actions d'instance `terminate` et `stop`. Si vous avez spécifié le comportement d'interruption d'instance `hibernate` lors de la demande de vos instances Spot ou de votre parc d'instances Spot, le drainage des instances Spot Amazon ECS n'est pas pris en charge pour ces instances.

Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur. Les tâches de service ayant l'état `PENDING` sur l'instance de conteneur faisant l'objet du drainage sont arrêtées immédiatement. S'il y a des instances de conteneur disponibles dans le cluster, des tâches de service de remplacement sont lancées dessus.

Vous pouvez activer le drainage d'instance Spot lorsque vous lancez une instance. Vous devez définir le paramètre `ECS_ENABLE_SPOT_INSTANCE_DRAINING` avant de démarrer l'agent de conteneur. Remplacez *`my-cluster`* par le nom de votre cluster.

```
[Environment]::SetEnvironmentVariable("ECS_ENABLE_SPOT_INSTANCE_DRAINING", "true",
 "Machine")

# Initialize the agent
Initialize-ECSAgent -Cluster my-cluster
```

Pour plus d'informations, consultez the section called "Lancement d'une instance de conteneur".

Clusters Amazon ECS pour instances externes

Amazon ECS Anywhere fournit un support pour l'enregistrement d'une Instance externe, telle qu'un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS. Les instances externes sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de processus. Si votre application nécessite du trafic entrant, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace. Amazon ECS a ajouté un nouveau type de lancement `EXTERNAL` que vous pouvez utiliser pour créer des services ou exécuter des tâches sur vos instances externes.

Systèmes d'exploitation et architectures système pris en charge

Voici la liste des systèmes d'exploitation pris en charge. Les architectures d'UC `x86_64` et `ARM64` sont prises en charge.

- Amazon Linux 2023
- Ubuntu 20, Ubuntu 22, Ubuntu 24
- RHEL 9 — Vous devez vous assurer que Docker est installé avant d'exécuter le Script d'installation d'ECS Anywhere. Pour plus d'informations, voir Installation du moteur Docker sur RHEL dans la documentation Docker.

À compter du 7 août 2026, les systèmes d'exploitation suivants ne sont plus pris en charge par Amazon ECS Anywhere :

- Amazon Linux 2
- CentOS Stream 9
- RHEL 7, RHEL 8
- Fedora 32, Fedora 33, Fedora 40
- openSUSE Tumbleweed
- Ubuntu 18
- Debian 9, Debian 10, Debian 11, Debian 12
- SUSE Enterprise Server 15
- Windows Server 2022, Windows Server 2019, Windows Server 2016, Windows Server 20H2

Considérations

Avant de démarrer l'utilisation des instances externes, tenez compte des considérations suivantes.

- Vous pouvez enregistrer une instance externe dans un cluster à la fois. Pour plus d'informations sur l'enregistrement d'une instance externe auprès d'un autre cluster, veuillez consulter Annulation de l’enregistrement d’une instance externe Amazon ECS.
- Vos instances externes ont besoin d'un rôle IAM qui leur permet de communiquer avec AWS API. Pour plus d’informations, consultez Rôle IAM Amazon ECS Anywhere.
- Vos instances externes ne doivent pas avoir de chaîne d'informations d'identification d'instance préconfigurée définie localement, car cela interférera avec le script d'enregistrement.
- Pour envoyer les journaux de conteneurs à CloudWatch Journaux, assurez-vous de créer et de spécifier un rôle IAM d'exécution de tâches dans votre définition de tâche.
- Lorsqu'une instance externe est enregistrée dans un cluster, la valeur `ecs.capability.external` est associée à l'instance. Cet attribut identifie l'instance en tant qu'instance externe. Vous pouvez ajouter des attributs personnalisés à vos instances externes pour les utiliser comme contrainte de placement de tâches. Pour plus d’informations, consultez Attributs personnalisés.
- Vous pouvez ajouter des balises de ressources à votre instance externe. Pour plus d’informations, consultez Ajout de balises à une instance de conteneur externe pour Amazon ECS.
- ECS Exec est pris en charge sur les instances externes. Pour plus d’informations, consultez Surveillance des conteneurs Amazon ECS avec ECS Exec.
- Vous trouverez ci-dessous des considérations supplémentaires spécifiques aux réseaux avec vos instances externes. Pour plus d’informations, consultez Mise en réseau.
  - La répartition de charge de service n'est pas prise en charge.
  - La découverte de service n'est pas prise en charge.
  - Les tâches qui s'exécutent sur des instances externes doivent utiliser les modes réseau `bridge`, `host` ou `none`. Le mode réseau `awsvpc` n'est pas pris en charge.
  - Il existe des domaines de service Amazon ECS dans chaque AWS Région. Ces domaines de service doivent être autorisés à envoyer du trafic vers vos instances externes.
  - Le SSM Agent installé sur votre instance externe gère les informations d'identification IAM qui effectuent une rotation toutes les 30 minutes à l'aide d'une empreinte matérielle. Si votre instance externe perd la connexion à AWS, l'agent SSM actualise automatiquement les informations d'identification une fois la connexion rétablie. Pour plus d'informations, consultez

Validation de serveurs et de machines virtuelles sur site à l'aide d'une empreinte matérielle dans le Guide de l'utilisateur AWS Systems Manager .

- Vous pouvez exécuter des tâches Linux sur des instances externes dans un IPv6-only configuration tant que les instances sont en IPv6-only sous-réseaux. Pour plus d'informations, consultez Utilisation d'un VPC dans IPv6-only mode.

- L'API `UpdateContainerAgent` n'est pas prise en charge. Pour obtenir des instructions sur la mise à jour du SSM Agent ou de l'agent Amazon ECS sur vos instances externes, veuillez consulter Mettre à jour le AWS Systems Manager agent et agent de conteneur Amazon ECS sur une instance externe.
- Les fournisseurs de capacité Amazon ECS ne sont pas pris en charge. Pour créer un service ou exécuter une tâche autonome sur vos instances externes, utilisez le type de lancement `EXTERNAL`.
- SELinux n'est pas pris en charge.
- L'utilisation de volumes Amazon EFS ou la spécification d'une `EFSVolumeConfiguration` n'est pas prise en charge.
- L'intégration à App Mesh n'est pas prise en charge.
- Si vous utilisez la console pour créer une définition de tâche d'instance externe, vous devez créer la définition de tâche avec l'éditeur JSON de la console.
- Lorsque vous utilisez un site qui n'est pas Amazon ECS-optimized AMI, exécutez les commandes suivantes sur l'instance de conteneur externe pour configurer les règles d'utilisation des rôles IAM pour les tâches. Pour plus d'informations, consultez Configuration supplémentaire d'instance externe.

```
$ sysctl -w net.ipv4.conf.all.route_localnet=1
$ iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
$ iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Mise en réseau

Les instances externes Amazon ECS sont optimisées pour exécuter des applications qui génèrent du trafic sortant ou des données de traitement. Si votre application nécessite un trafic entrant, tel qu'un service web, l'absence de prise en charge d'Elastic Load Balancing rend l'exécution de ces applications moins efficace, car il n'existe pas de prise en charge pour placer ces charges de travail derrière un équilibreur de charge.

Vous trouverez ci-dessous des considérations supplémentaires spécifiques aux réseaux avec vos instances externes.

- La répartition de charge de service n'est pas prise en charge.
- La découverte de service n'est pas prise en charge.
- Les tâches Linux qui s'exécutent sur des instances externes doivent utiliser les modes réseau `bridge`, `host` ou `none`. Le mode réseau `awsvpc` n'est pas pris en charge.

  Pour savoir comment opérer un réseau, consultez la section Options de mise en réseau des tâches Amazon ECS pour les instances EC2.
- Vous pouvez exécuter des tâches Linux sur des instances externes dans un IPv6-only configuration tant que les instances sont en IPv6-only sous-réseaux. Pour plus d'informations, consultez Utilisation d'un VPC dans IPv6-only mode.
- Il existe des domaines de service Amazon ECS dans chaque région et vous devez être autorisé à envoyer du trafic vers vos instances externes.
- Le SSM Agent installé sur votre instance externe gère les informations d'identification IAM qui effectuent une rotation toutes les 30 minutes à l'aide d'une empreinte matérielle. Si votre instance externe perd la connexion à AWS, l'agent SSM actualise automatiquement les informations d'identification une fois la connexion rétablie. Pour plus d'informations, consultez Validation de serveurs et de machines virtuelles sur site à l'aide d'une empreinte matérielle dans le Guide de l'utilisateur AWS Systems Manager .

Les domaines suivants sont utilisés pour la communication entre le service Amazon ECS service et l'agent Amazon ECS installé sur votre instance externe. Assurez-vous que le trafic est autorisé et que la résolution DNS fonctionne. Pour chaque point de terminaison, *`region`* représente l'identifiant de région pour un AWS Région prise en charge par Amazon ECS, telle que `us-east-2` pour la région de l'Est des États-Unis (Ohio). Les points de terminaison de toutes les régions que vous utilisez doivent être autorisés. Pour les points de terminaison `ecs-a` et `ecs-t`, vous devez inclure un astérisque (par exemple, `ecs-a-*`).

- `ecs-a-*.`*`region`*`.amazonaws.com` — Ce point de terminaison est utilisé lors de la gestion des tâches.
- `ecs-t-*.`*`region`*`.amazonaws.com` — Ce point de terminaison est utilisé pour gérer les métriques de tâche et de conteneur.
- `ecs.`*`region`*`.amazonaws.com` — Il s'agit du point de terminaison de service pour Amazon ECS.

- `ssm.region.amazonaws.com` — Il s'agit du point de terminaison de service pour AWS Systems Manager.
- `ec2messages.region.amazonaws.com` — Il s'agit du point de terminaison du service qui AWS Systems Manager permet de communiquer entre l'agent Systems Manager et le service Systems Manager dans le cloud.
- `ssmmessages.region.amazonaws.com` — Il s'agit du point de terminaison requis pour créer et supprimer des canaux de session avec le service Session Manager dans le cloud.
- Si vos tâches nécessitent une communication avec d'autres AWS services, assurez-vous que ces points de terminaison de service sont autorisés. Les exemples d'applications incluent l'utilisation d'Amazon ECR pour extraire des images de conteneurs ou l'utilisation de CloudWatch pour CloudWatch Journaux. Pour plus d'informations, consultez Points de terminaison de service dans la Référence générale AWS .

Amazon FSx for Windows File Server avec ECS Anywhere

> **Important**
>
> La prise en charge d'Amazon ECS Anywhere par Windows est obsolète. Cette section n'est plus applicable.

Afin d'utiliser le Amazon FSx for Windows File Server avec les instances externes Amazon ECS, vous devez établir une connexion entre votre centre de données sur site et AWS Cloud. Pour plus d'informations sur la connexion de votre réseau à votre VPC, consultez Options de connectivité du cloud privé virtuel Amazon.

gMSA avec ECS Anywhere

> **Important**
>
> La prise en charge d'Amazon ECS Anywhere par Windows est obsolète. Cette section n'est plus applicable.

Les cas d'utilisation suivants étaient pris en charge pour ECS Anywhere lorsque Windows était un système d'exploitation pris en charge.

- L'Active Directory se trouve dans AWS Cloud - Pour cette configuration, vous créez une connexion entre votre réseau local et AWS Cloud à l'aide d'un AWS Direct Connect connexion. Pour en savoir plus comment créer une connexion, consultez Options de connectivité du cloud privé virtuel Amazon. Vous pouvez créer un Active Directory dans AWS Cloud. Pour plus d'informations sur la façon de démarrer avec AWS Directory Service, voir Configuration AWS Directory Service dans le AWS Directory Service Guide d'administration. Vous pouvez ensuite joindre vos instances externes au domaine à l'aide du AWS Direct Connect connexion. Pour en savoir plus sur l'utilisation de gMSA avec Amazon ECS, consultez the section called “Utilisation des gMSA les gMSA pour les conteneurs Windows EC2”.
- Active Directory se trouve dans le centre de données sur site. - Pour cette configuration, vous joignez vos instances externes à Active Directory sur site. Vous utilisez ensuite les informations d'identification disponibles localement lorsque vous exécutez les tâches Amazon ECS.

Création d’un cluster Amazon ECS pour les charges de travail d’instance externe

Vous créez un cluster pour définir l’infrastructure sur laquelle vos tâches et vos services s’exécutent.

Avant de commencer, veillez achever les étapes de Configurer l'utilisation d'Amazon ECS et affectez l’autorisation IAM appropriée. Pour plus d’informations, consultez the section called “Exemples de cluster Amazon ECS”. La console Amazon ECS fournit un moyen simple de créer les ressources nécessaires à un cluster Amazon ECS en créant un CloudFormation empiler.

Pour que le processus de création de cluster soit aussi simple que possible, la console dispose de sélections par défaut pour de nombreux choix que nous décrivons ci-dessous. La plupart des sections de la console comportent des volets d'aide qui fournissent un contexte supplémentaire.

Vous pouvez modifier les options suivantes :

- Ajoutez un espace de noms au cluster.

  Un espace de noms permet aux services que vous créez dans le cluster de se connecter aux autres services de l'espace de noms sans configuration supplémentaire. Pour plus d’informations, consultez Interconnexion des services Amazon ECS.
- Configuration du cluster pour les instances externes

- Attribuer un AWS KMS clé pour votre stockage géré. Pour plus d'informations sur la création d'une clé, consultez la section Création d'une clé KMS dans le Guide de l'utilisateur AWS Key Management Service .
- Ajoutez des balises pour vous aider à identifier vos clusters.

Pour créer un nouveau cluster (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters, choisissez Create Cluster (Créer un cluster).
5. Sous Configuration de cluster, configurez les éléments suivants :
   - Pour Nom du cluster, saisissez un nom unique.

     Le nom peut contenir jusqu'à 255 lettres (minuscules et majuscules), des chiffres et des traits d'union.
   - (Facultatif) Pour que l'espace de noms utilisé pour Service Connect soit différent du nom du cluster, saisissez un nom unique dans Espace de nom.
6. (Facultatif) Utilisez Container Insights, développez Surveillance, puis sélectionnez une des options suivantes :
   - Pour utiliser Container Insights avec observabilité améliorée comme recommandé, sélectionnez Container Insights avec observabilité améliorée.
   - Pour utiliser Container Insights, choisissez Container Insights.
7. (Facultatif) Pour utiliser ECS Exec afin de déboguer les tâches dans le cluster, développez Configuration de la résolution des problèmes, puis configurez les éléments suivants :
   - Sélectionnez Activer sur ECS Exec.
   - (Facultatif) Pour AWS KMS clé pour ECS Exec, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de session ECS Exec.
   - (Facultatif) Pour Journalisation ECS Exec, choisissez la destination du journal :
     - Pour envoyer des journaux à CloudWatch Journaux, choisissez Amazon CloudWatch.
     - Pour envoyer les journaux à Amazon S3, choisissez Amazon S3.
     - Pour désactiver la journalisation, choisissez Aucune.

8. (Facultatif) Chiffrez les données sur le stockage géré. En dessous Chiffrement, pour Stockage géré, entrez l'ARN du AWS KMS clé que vous souhaitez utiliser pour chiffrer les données de stockage gérées.
9. (Facultatif) Pour vous aider à identifier votre cluster, développez Tags (balises), puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

   - Pour Key (Clé), saisissez le nom de la clé.
   - Pour Valeur, saisissez la valeur de clé.
10. Choisissez Créer.

Étapes suivantes

Vous devez enregistrer les instances auprès du cluster. Pour plus d'informations, consultez Enregistrement d'une instance externe dans un cluster Amazon ECS.

Créez une définition de tâche pour le type de lancement externe. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console

Exécutez vos applications en tant que tâches autonomes ou dans le cadre d'un service. Pour plus d'informations, consultez les ressources suivantes :

- Exécution d'une application en tant que tâche Amazon ECS
- Création d'un déploiement de mise à jour propagée Amazon ECS

Enregistrement d'une instance externe dans un cluster Amazon ECS

Pour chaque instance externe que vous enregistrez auprès d'un cluster Amazon ECS, le SSM Agent, l'agent de conteneur Amazon ECS et Docker doivent être installés. Pour enregistrer l'instance externe dans un cluster Amazon ECS, elle doit d'abord être enregistrée en tant que AWS Systems Manager instance gérée. Vous pouvez créer le script d'installation en quelques clics sur la console Amazon ECS. Le script d'installation inclut une clé d'activation Systems Manager et des commandes pour installer chacun des agents requis et Docker. Le script d'installation doit être exécuté sur votre serveur local ou votre machine virtuelle pour terminer la procédure d'installation et d'enregistrement.

> **Note**
>
> Avant d'enregistrer votre instance externe Linux auprès du cluster, créez le fichier `/etc/ecs/ecs.config` sur votre instance externe et ajoutez les paramètres de configuration de l'agent de conteneur souhaités. Vous ne pouvez pas le faire après l'enregistrement de l'instance externe dans un cluster. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Dans la page Clusters, sélectionnez un cluster dans lequel enregistrer votre instance externe.
5. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet.
6. Sur la page Register external instances (Enregistrer des instances externes), effectuez les étapes suivantes.

    a. Pour Activation key duration (in days) [Durée de la clé d'activation (en jours)], saisissez le nombre de jours pendant lesquels la clé d'activation reste active. Une fois que le nombre de jours que vous avez saisis est écoulé, la clé ne fonctionne plus lors de l'enregistrement d'une instance externe.

    b. Pour Number of instances (Nombre d'instances), saisissez le nombre d'instances externes que vous souhaitez enregistrer dans votre cluster avec la clé d'activation.

    c. Pour Instance role (Rôle d'instance), choisissez le rôle IAM à associer à vos instances externes. Si aucun rôle n'a déjà été créé, choisissez Create new role (Créer un rôle) Pour qu'Amazon ECS crée un rôle en votre nom. Pour plus d'informations sur les autorisations IAM requises pour vos instances externes, consultez Rôle IAM Amazon ECS Anywhere.

    d. Copiez la commande d'inscription. Cette commande doit être exécutée sur chaque instance externe que vous souhaitez enregistrer dans le cluster.

> **Important**
>
> La partie bash du script doit être exécutée en tant que root. Si la commande n'est pas exécutée en tant que racine, une erreur est renvoyée.

e. Choisissez Close (Fermer).

AWS CLI for Linux operating systems

1. Créez une paire d'activations de Systems Manager. Elle est utilisée pour l'activation de l'instance gérée par Systems Manager. La sortie inclut un `ActivationId` et un `ActivationCode`. Vous les utiliserez dans une étape ultérieure. Veillez à spécifier le rôle IAM ECS Anywhere que vous avez créé. Pour plus d'informations, consultez Rôle IAM Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Sur votre serveur local ou votre machine virtuelle (VM), téléchargez le script d'installation.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh"
```

3. (Facultatif) Sur votre serveur sur site ou votre machine virtuelle (VM), procédez comme suit pour vérifier le script d'installation à l'aide du fichier de signature du script.

   a. Téléchargez et installez GnuPG. Pour plus d'informations sur GNUpg, consultez le site Web GnuPG. Pour les systèmes Linux, installez `gpg` à l'aide du gestionnaire de package de votre version de Linux.

   b. Extrayez la clé publique PGP Amazon ECS.

```
gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
```

   c. Téléchargez la signature du script d'installation. La signature est une signature PGP détachées ASCII, stockée dans un fichier portant l'extension `.asc`.

```
curl --proto "https" -o "/tmp/ecs-anywhere-install.sh.asc" "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install-latest.sh.asc"
```

d. Vérifiez le fichier de script d'installation à l'aide de la clé.

```
gpg --verify /tmp/ecs-anywhere-install.sh.asc /tmp/ecs-anywhere-install.sh
```

La sortie attendue est la suivante :

```
gpg: Signature made Tue 25 May 2021 07:16:29 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the
 owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Sur votre serveur sur site ou votre machine virtuelle (VM), exécutez le script d'installation. Spécifiez le nom du cluster, la région, l'ID d'activation de Systems Manager et le code d'activation à partir de la première étape.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE
```

Pour un serveur local ou une machine virtuelle (VM) sur lequel le pilote NVIDIA est installé pour les charges de travail GPU, vous devez ajouter l'indicateur `--enable-gpu` du script d'installation. Lorsque cet indicateur est spécifié, le script d'installation vérifie que le pilote NVIDIA est en cours d'exécution, puis ajoute les variables de configuration requises pour exécuter vos tâches Amazon ECS. Pour plus d'informations sur l'exécution de charges de travail GPU et la spécification des exigences GPU dans une définition de tâche, consultez Spécification des GPU dans une définition de tâche Amazon ECS.

```
sudo bash /tmp/ecs-anywhere-install.sh \
    --region $REGION \
    --cluster $CLUSTER_NAME \
    --activation-id $ACTIVATION_ID \
    --activation-code $ACTIVATION_CODE \
    --enable-gpu
```

Pour enregistrer une instance externe existante auprès d'un cluster différent, procédez comme suit.

Pour enregistrer une instance externe existante auprès d'un cluster différent

1. Arrêtez l'agent de conteneur Amazon ECS.

```
sudo systemctl stop ecs.service
```

2. Modifiez le fichier `/etc/ecs/ecs.config` et sur la ligne `ECS_CLUSTER`, assurez-vous que le nom du cluster correspond au nom du cluster auprès duquel vous souhaitez enregistrer l'instance externe.
3. Supprimez les données existantes de l'agent Amazon ECS.

```
sudo rm /var/lib/ecs/data/agent.db
```

4. Démarrez l'agent de conteneur Amazon ECS.

```
sudo systemctl start ecs.service
```

AWS CLI for Windows operating systems

1. Créez une paire d'activations de Systems Manager. Elle est utilisée pour l'activation de l'instance gérée par Systems Manager. La sortie inclut un `ActivationId` et un `ActivationCode`. Vous les utiliserez dans une étape ultérieure. Veillez à spécifier le rôle IAM ECS Anywhere que vous avez créé. Pour plus d'informations, consultez Rôle IAM Amazon ECS Anywhere.

```
aws ssm create-activation --iam-role ecsAnywhereRole | tee ssm-activation.json
```

2. Sur votre serveur local ou votre machine virtuelle (VM), téléchargez le script d'installation.

```
Invoke-RestMethod -URI "https://amazon-ecs-agent.s3.amazonaws.com/ecs-anywhere-install.ps1" -OutFile “ecs-anywhere-install.ps1”
```

3. (Facultatif) Le script PowerShell est signé par Amazon et, par conséquent, Windows effectue automatiquement la validation du certificat sur ce dernier. Aucune validation manuelle n'est requise.

Pour vérifier manuellement le certificat, effectuez un clic droit sur le fichier, accédez aux propriétés et utilisez l'onglet Signatures numériques pour obtenir plus de détails.

Cette option n'est disponible que lorsque l'hôte possède le certificat dans le magasin de certificats.

La vérification doit renvoyer des informations semblables à ce qui suit :

```
# Verification (PowerShell)
Get-AuthenticodeSignature -FilePath .\ecs-anywhere-install.ps1

SignerCertificate                         Status      Path
-----------------                         ------      ----
EXAMPLECERTIFICATE                        Valid       ecs-anywhere-install.ps1

...

Subject              : CN="Amazon Web Services, Inc.",...

----
```

4. Sur votre serveur sur site ou votre machine virtuelle (VM), exécutez le script d'installation. Spécifiez le nom du cluster, la région, l'ID d'activation de Systems Manager et le code d'activation à partir de la première étape.

```
.\ecs-anywhere-install.ps1 -Region $Region -Cluster $Cluster -
ActivationID $ActivationID -ActivationCode $ActivationCode
```

5. Vérifiez que l'agent de conteneur Amazon ECS est en cours d'exécution.

```
Get-Service AmazonECS

Status   Name               DisplayName
------   ----               -----------
Running  AmazonECS          Amazon ECS
```

Pour enregistrer une instance externe existante auprès d'un cluster différent, procédez comme suit.

Pour enregistrer une instance externe existante auprès d'un cluster différent

1. Arrêtez l'agent de conteneur Amazon ECS.

```
Stop-Service AmazonECS
```

2. Modifier le paramètre `ECS_CLUSTER` afin que le nom du cluster corresponde au nom du cluster auprès duquel vous souhaitez enregistrer l'instance externe.

```
[Environment]::SetEnvironmentVariable("ECS_CLUSTER", $ECSCluster,
 [System.EnvironmentVariableTarget]::Machine)
```

3. Supprimez les données existantes de l'agent Amazon ECS.

```
Remove-Item -Recurse -Force $env:ProgramData\Amazon\ECS\data\*
```

4. Démarrez l'agent de conteneur Amazon ECS.

```
Start-Service AmazonECS
```

Le AWS CLI peut être utilisé pour créer une activation de Systems Manager avant d'exécuter le script d'installation afin de terminer le processus d'enregistrement de l'instance externe.

Annulation de l’enregistrement d’une instance externe Amazon ECS

Nous vous recommandons de désenregistrer l'instance auprès d'Amazon ECS et AWS Systems Manager une fois que vous avez terminé avec l'instance. Après l'annulation de l'enregistrement, l'instance externe n'est plus en mesure d'accepter de nouvelles tâches.

Si des tâches sont en cours d'exécution sur l'instance de conteneur lorsque vous annulez l'enregistrement, ces tâches restent en cours d'exécution jusqu'à ce que vous les arrêtiez d'une autre manière. Toutefois, ces tâches ne sont plus surveillées ou prise en compte par Amazon ECS. Si ces tâches sur votre instance externe font partie d'un Amazon ECS service, le planificateur de service commence une autre copie de cette tâche sur une autre instance de conteneur, si possible.

Après avoir désenregistré l'instance, nettoyez le reste AWS ressources relatives à l'instance. Vous pouvez ensuite l’enregistrer dans un nouveau cluster.

Procédure

Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance externe.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet.
5. Sous Constainer instances (Instances de conteneur), sélectionnez l'ID de l'instance externe pour annuler l'enregistrement. Vous êtes redirigé vers la page de détails de l'instance de conteneur.
6. Sur le Instance de conteneur : ***id*** page, choisissez Désenregistrer.
7. Passez en revue le message d'annulation d'enregistrement. Sélectionnez Deregister from AWS Systems Manager(Annuler l'enregistrement AWS Systems Manager) pour également annuler l'enregistrement de l'instance externe en tant qu'instance gérée par Systems Manager. Choisissez Deregister (Annuler l'enregistrement).

   > **Note**
   >
   > Vous pouvez annuler l'enregistrement de l'instance externe en tant qu'instance gérée par Systems Manager dans la console Systems Manager. Pour plus d'informations, consultez la section Annuler l'enregistrement des nœuds gérés dans un environnement hybride et multicloud dans le Guide de l'utilisateur AWS Systems Manager .

8. Après avoir désenregistré l'instance, nettoyez AWS ressources sur votre serveur ou machine virtuelle sur site.

Système d'exploitation	Étapes
Linux	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.

Système d’exploitation	Étapes	
	`sudo systemctl stop ecs amazon-ssm-agent`	

	b. Supprimez les packages Amazon ECS et Systems Manager.
	
	Pour CentOS 7, CentOS 8 et RHEL 7
	
	`sudo yum remove -y amazon-ecs-init amazon-ssm-agent`
	
	Pour SUSE Enterprise Server 15
	
	`sudo zypper remove -y amazon-ecs-init amazon-ssm-agent`
	
	Pour Debian et Ubuntu
	
	`sudo apt remove -y amazon-ecs-init amazon-ssm-agent`	

	c. Supprimez les répertoires restants. `sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Système d'exploitation	Étapes

Windows	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Supprimez le package Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`

AWS CLI

1. Vous avez besoin de l'ID d'instance et de l'ARN de l'instance de conteneur pour annuler l'enregistrement de l'instance de conteneur. Si vous n'avez pas ces valeurs, exécutez les commandes suivantes.

   Exécutez la commande suivante pour obtenir l'ID d'instance.

   Vous utilisez l'ID d'instance (`instanceID`) pour obtenir l'ARN de l'instance de conteneur (`containerInstanceARN`).

   ```
   instanceId=$(aws ssm describe-instance-information --region "{{ region }}" |
    jq ".InstanceInformationList[] |select(.IPAddress==\"{{ IPv4 Address }}\")
    | .InstanceId" | tr -d'"'
   ```

   Exécutez les commandes suivantes.

Vous utilisez l'`containerInstanceArn` comme paramètre dans la commande pour annuler l'enregistrement de l'instance (`deregister-container-instance`).

```
instances=$(aws ecs list-container-instances --cluster "{{ cluster }}" --region
 "{{ region }}" | jq -c '.containerInstanceArns')
containerInstanceArn=$(aws ecs describe-container-instances --cluster
 "{{ cluster }}" --region "{{ region }}" --container-instances $instances
 | jq ".containerInstances[] | select(.ec2InstanceId==\"{{ instanceId }}\")
 | .containerInstanceArn" | tr -d '"')
```

2. Exécutez la commande suivante pour purger l'instance.

```
aws ecs update-container-instances-state --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instances "{{ containerInstanceArn }}" --status
 DRAINING
```

3. Une fois le drainage de l'instance de conteneur terminé, exécutez la commande suivante pour annuler son enregistrement.

```
aws ecs deregister-container-instance --cluster "{{ cluster }}" --region
 "{{ region }}" --container-instance "{{ containerInstanceArn }}"
```

4. Exécutez la commande suivante pour supprimer l'instance de conteneur de SSM.

```
aws ssm deregister-managed-instance --region "{{ region }}" --instance-id
 "{{ instanceId }}"
```

5. Après avoir désenregistré l'instance, nettoyez AWS ressources sur votre serveur ou machine virtuelle sur site.

Système d’exploitation	Étapes	
Linux	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.	

Système d’exploitation	Étapes	
	`sudo systemctl stop ecs amazon-ssm-agent`	
	b. Supprimez les packages Amazon ECS et Systems Manager. `sudo (yum/apt/zypper) remove amazon-ecs-init amazon-ssm-agent`	
	c. Supprimez les répertoires restants. `sudo rm -rf /var/lib/ecs /etc/ecs /var/lib/amazon/ssm /var/log/ecs /var/log/amazon/ssm`	

Système d'exploitation	Étapes	

Windows	a. Arrêtez l'agent de conteneur Amazon ECS et les services SSM Agent sur l'instance.
	`Stop-Service AmazonECS`
	`Stop-Service AmazonSSMAgent`
	b. Supprimez le package Amazon ECS.
	`.\ecs-anywhere-install.ps1 -Uninstall`	

Mettre à jour le AWS Systems Manager agent et agent de conteneur Amazon ECS sur une instance externe

Votre serveur local ou votre machine virtuelle doit exécuter à la fois AWS Systems Manager Agent (agent SSM) et agent de conteneur Amazon ECS lors de l'exécution de charges de travail Amazon ECS. AWS publie de nouvelles versions de ces agents lorsque des fonctionnalités sont ajoutées ou mises à jour. Si vos instances externes utilisent une version antérieure de l'un ou l'autre des agents, vous pouvez les mettre à jour à l'aide des procédures suivantes.

Mise à jour de SSM Agent sur une instance externe

AWS Systems Manager vous recommande d'automatiser le processus de mise à jour de l'agent SSM sur vos instances. Ils fournissent plusieurs méthodes pour automatiser les mises à jour. Pour de plus amples informations, veuillez consulter Automatisation des mises à jour sur SSM Agent dans le Guide de l'utilisateur AWS Systems Manager .

Mise à jour de l'agent Amazon ECS sur une instance externe

Sur vos instances externes, l'agent de conteneur Amazon ECS est mis à jour en mettant à niveau le package `ecs-init`. La mise à jour de l'agent Amazon ECS n'interrompt pas les tâches ou les services en cours d'exécution. Amazon ECS fournit le package `ecs-init` et le fichier SIGNATURE dans un compartiment Amazon S3 dans chaque région. En commençant par `ecs-init` version `1.52.1-1`, Amazon ECS fournit des packages `ecs-init` distincts à utiliser en fonction du système d'exploitation et de l'architecture système que votre instance externe utilise.

Utilisez le tableau suivant pour déterminer le package `ecs-init` que vous devez télécharger en fonction du système d'exploitation et de l'architecture système que votre instance externe utilise.

> **Note**
>
> Vous pouvez déterminer le système d'exploitation et l'architecture système que votre instance externe utilise à l'aide des commandes suivantes.
>
> ```
> cat /etc/os-release
> uname -m
> ```

Systèmes d'exploitation (architecture)	package ecs-init
CentOS 7 (x86_64) CentOS 8 (x86_64) CentOS Stream 9 (x86_64) SUSE Enterprise Server 15 (x86_64) RHEL 7 (x86_64) RHEL 8 (x86_64)	`amazon-ecs-init-latest.x86_64.rpm`
CentOS 7 (aarch64) CentOS 8 (aarch64) CentOS Stream 9 (aarch64)	`amazon-ecs-init-latest.aarch64.rpm`

Systèmes d'exploitation (architecture)	package ecs-init
RHEL 7 (aarch64)	
Debian 9 (x86_64) Debian 10 (x86_64) Debian 11 (x86_64) Debian 12 (x86_64) Ubuntu 18 (x86_64) Ubuntu 20 (x86_64) Ubuntu 22 (x86_64) Ubuntu 24 (x86_64)	`amazon-ecs-init-latest.amd64.deb`
Debian 9 (aarch64) Debian 10 (aarch64) Debian 11 (aarch64) Debian 12 (aarch64) Ubuntu 18 (aarch64) Ubuntu 20 (aarch64) Ubuntu 22 (aarch64) Ubuntu 24 (aarch64)	`amazon-ecs-init-latest.arm64.deb`

Procédez comme suit pour mettre à jour l'agent Amazon ECS.

Pour mettre à jour l'agent Amazon ECS

1. Confirmez la version de l'agent Amazon ECS que vous exécutez.

```
curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
```

2. Télécharger le package `ecs-init` pour votre système d'exploitation et votre architecture système. Amazon ECS fournit le fichier de package `ecs-init` dans un compartiment Amazon S3 dans chaque région. Assurez-vous de remplacer le *<region>* identifiant dans la commande avec le nom de la région (par exemple, `us-west-2`) dont vous êtes géographiquement le plus proche.

   amazon-ecs-init-latest.x86_64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm
   ```

   amazon-ecs-init-latest.aarch64.rpm

   ```
   curl -o amazon-ecs-init.rpm https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm
   ```

   amazon-ecs-init-latest.amd64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb
   ```

   amazon-ecs-init-latest.arm64.deb

   ```
   curl -o amazon-ecs-init.deb https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb
   ```

3. (Facultatif) Vérification de la validité du fichier de package `ecs-init` à l'aide de la signature PGP.

   a. Téléchargez et installez GnuPG. Pour plus d'informations sur GNUpg, consultez le site Web GnuPG. Pour les systèmes Linux, installez gpg à l'aide du gestionnaire de package de votre version de Linux.

   b. Extrayez la clé publique PGP Amazon ECS.

      ```
      gpg --keyserver hkp://keys.gnupg.net:80 --recv BCE9D9A42D51784F
      ```

c. Téléchargez la signature de package `ecs-init`. La signature est une signature PGP détachées ASCII, stockée dans un fichier portant l'extension `.asc`. Amazon ECS fournit le fichier SIGNATURE dans un compartiment Amazon S3 dans chaque région. Assurez-vous de remplacer le *<region>* identifiant dans la commande avec le nom de la région (par exemple, `us-west-2`) dont vous êtes géographiquement le plus proche.

   amazon-ecs-init-latest.x86_64.rpm

   ```
   curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.x86_64.rpm.asc
   ```

   amazon-ecs-init-latest.aarch64.rpm

   ```
   curl -o amazon-ecs-init.rpm.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.aarch64.rpm.asc
   ```

   amazon-ecs-init-latest.amd64.deb

   ```
   curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.amd64.deb.asc
   ```

   amazon-ecs-init-latest.arm64.deb

   ```
   curl -o amazon-ecs-init.deb.asc https://s3.<region>.amazonaws.com/amazon-ecs-agent-<region>/amazon-ecs-init-latest.arm64.deb.asc
   ```

d. Vérifiez le fichier de package `ecs-init` à l'aide de la clé.

   Pour les packages **`rpm`**

   ```
   gpg --verify amazon-ecs-init.rpm.asc ./amazon-ecs-init.rpm
   ```

   Pour les packages **`deb`**

   ```
   gpg --verify amazon-ecs-init.deb.asc ./amazon-ecs-init.deb
   ```

   La sortie attendue est la suivante :

```
gpg: Signature made Fri 14 May 2021 09:31:36 PM UTC
gpg:                using RSA key 50DECCC4710E61AF
gpg: Good signature from "Amazon ECS <ecs-security@amazon.com>" [unknown]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: F34C 3DDA E729 26B0 79BE  AEC6 BCE9 D9A4 2D51 784F
     Subkey fingerprint: D64B B6F9 0CF3 77E9 B5FB  346F 50DE CCC4 710E 61AF
```

4. Installez le package `ecs-init`.

   Pour le package **`rpm`** sur CentOS 7, CentOS 8 et RHEL 7

   ```
   sudo yum install -y ./amazon-ecs-init.rpm
   ```

   Pour le package **`rpm`** sur SUSE Enterprise Server 15

   ```
   sudo zypper install -y --allow-unsigned-rpm ./amazon-ecs-init.rpm
   ```

   Pour le package **`deb`**

   ```
   sudo dpkg -i ./amazon-ecs-init.deb
   ```

5. Redémarrez le service `ecs`.

   ```
   sudo systemctl restart ecs
   ```

6. Vérifiez que la version d'agent Amazon ECS a été mise à jour.

   ```
   curl -s 127.0.0.1:51678/v1/metadata | python3 -mjson.tool
   ```

Mise à jour d’un cluster Amazon ECS.

Vous pouvez modifier les propriétés de cluster suivantes :

- Définissez un fournisseur de capacité par défaut.

  Chaque cluster dispose d'un ou plusieurs fournisseurs de capacité et éventuellement d'une stratégie de fournisseur de capacité facultative. La stratégie de fournisseur de capacité détermine

la façon dont les tâches sont réparties entre les fournisseurs de capacité du cluster. Lorsque vous exécutez une tâche autonome ou que vous créez un service, vous pouvez utiliser soit la stratégie de fournisseur de capacité par défaut du cluster, soit une stratégie de fournisseur de capacité qui remplace celle par défaut.

- Activez Container Insights.

  CloudWatch Container Insights collecte, agrège et résume les métriques et les journaux de vos applications conteneurisées et de vos microservices. Conteneur Insights fournit également des informations de diagnostic (par exemple sur les échecs de redémarrage des conteneurs) que vous pouvez utiliser pour isoler les problèmes et les résoudre rapidement. Pour plus d'informations, consultez the section called "Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée".
- Ajoutez des étiquettes pour vous aider à identifier vos clusters.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Mettre à jour le cluster.
5. Pour définir le fournisseur de capacité par défaut, sous Stratégie de fournisseur de capacité par défaut, sélectionnez Ajouter plus.

   a. Pour Fournisseur de capacité, choisissez le fournisseur de capacité.

   b. (Facultatif) Pour Base, saisissez le nombre minimal de tâches exécutées sur le fournisseur de capacité.

      Vous ne pouvez définir une valeur de Base que pour un seul fournisseur de capacité.

   c. (Facultatif) Pour Poids, saisissez le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié.

   d. (Facultatif) Répétez les étapes pour tous les fournisseurs de capacité supplémentaires.
6. Pour activer ou désactiver Container Insights, développez Surveillance, puis activez Utiliser Container Insights.
7. Pour vous aider à identifier votre cluster, développez Balises, puis configurez vos balises.

   [Add a tag] Choisissez Add tag (Ajouter une balise) et procédez comme suit :

- Pour Key (Clé), saisissez le nom de la clé.
- Pour Value (Valeur), saisissez la valeur de clé.

[Remove a tag] Choisissez Remove (Supprimer) à la droite de la clé et de la valeur de la balise.

8. Choisissez Mettre à jour.

Suppression d’un cluster Amazon ECS.

Si vous avez terminé d'utiliser un cluster, vous pouvez le supprimer. Une fois que vous avez supprimé le cluster, il passe à l'état `INACTIVE`. Les clusters ayant un état `INACTIVE` peuvent rester détectables dans votre compte pendant un certain temps. Cependant, cela est susceptible de changer à terme. Il est donc important de ne pas compter sur la persistance des clusters `INACTIVE`.

Avant de supprimer un cluster, vous devez effectuer les opérations suivantes :

- Supprimez tous les services du cluster. Pour plus d’informations, consultez the section called “Suppression d'un service”.
- Arrêtez toutes les tâches en cours d'exécution. Pour plus d’informations, consultez the section called “Arrêt d'une tâche”.
- Annulez l'enregistrement de toutes les instances de conteneur enregistrées dans le cluster. Pour plus d’informations, consultez the section called “Annulation de l'enregistrement d'une instance de conteneur”.
- Supprimez l'espace de noms. Pour plus d'informations, consultez Suppression des espaces de noms dans le Guide du développeur AWS Cloud Map .

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez la région à utiliser.
3. Dans le panneau de navigation, choisissez Clusters.
4. Sur la page Clusters sélectionnez le cluster à supprimer.
5. Choisissez Delete Cluster (Supprimer le cluster).

Un message s'affiche lorsque vous n'avez pas supprimé toutes les ressources associées au cluster.

6. Dans la case de confirmation, entrez supprimer ***cluster name***.

Annulation de l’enregistrement d’instance de conteneur Amazon ECS

> **Important**
>
> Cette rubrique concerne uniquement les instances de conteneur créées dans Amazon EC2. Pour plus d'informations sur l'annulation de l'enregistrement des instances externes, veuillez consulterAnnulation de l’enregistrement d’une instance externe Amazon ECS.

Lorsque vous avez terminé avec une instance de conteneur sauvegardée par Amazon EC2, vous pouvez en annuler l'enregistrement dans votre cluster. Après l'annulation de l'enregistrement, l'instance de conteneur n'est plus en mesure d'accepter de nouvelles tâches.

Si des tâches sont en cours d'exécution sur l'instance du conteneur lorsque vous annulez l'enregistrement, ces tâches restent en cours d'exécution jusqu'à ce que vous résiliez l'instance ou les tâches d'une autre manière. Toutefois, ces tâches sont orphelines, ce qui signifie qu'elles ne sont plus surveillées ou prise en compte par Amazon ECS. Si une tâche orpheline sur votre instance de conteneur fait partie d'un service Amazon ECS service, le planificateur de service commence une autre copie de cette tâche sur une autre instance de conteneur, si possible. L'enregistrement de tout conteneur des tâches de service orphelines enregistré avec un groupe cible Application Load Balancer est annulé. Le drainage de la connexion commence en fonction des paramètres de l'équilibreur de charge ou du groupe cible. Si une tâche orpheline utilise le mode réseau `awsvpc`, leurs interfaces réseau Elastic sont supprimées.

Si vous prévoyez d'utiliser l'instance de conteneur à d'autres fins après l'annulation de l'enregistrement, vous devez arrêter toutes les tâches en cours d'exécution sur l'instance de conteneur avant l'annulation de l'enregistrement. Vous éviterez ainsi que les tâches orphelines consomment des ressources.

Lors de la désinscription d'une instance de conteneur, tenez compte des considérations suivantes.

- Étant donné que chaque instance de conteneur possède des informations d'état uniques, elles ne peuvent pas faire l'objet d'une annulation d'enregistrement dans un cluster et d'un réenregistrement dans un autre. Pour déplacer des ressources d'instance de conteneur, nous vous recommandons de résilier les instances de conteneur d'un cluster et d'en lancer de nouvelles dans le nouveau cluster. Pour de plus amples informations, consultez la section Résilier votre instance dans le Guide de l'utilisateur Amazon EC2 et Lancement d'une instance de conteneur Amazon ECS Linux.
- Si l'instance de conteneur est gérée par un groupe Auto Scaling ou un CloudFormation pile, terminez l'instance en mettant à jour le groupe Auto Scaling ou CloudFormation pile. Sinon, le groupe Auto Scaling ou CloudFormation créera une instance après sa résiliation.
- Si vous résiliez une instance de conteneur en cours d'exécution avec un agent de conteneur Amazon ECS connecté, l'agent annule automatiquement l'enregistrement de l'instance dans votre cluster. L'annulation de l'enregistrement des instances de conteneur arrêtées ou des instances ayant des agents déconnectés n'est pas automatique lorsque ces instances sont résiliées.
- L'annulation de l'enregistrement d'une instance de conteneur supprime l'instance d'un cluster, mais ne résilie pas l'instance Amazon EC2. Si vous avez fini d'utiliser l'instance, veillez à la résilier pour arrêter la facturation. Pour de plus amples informations, veuillez consulter Résilier une instance dans le Guide de l'utilisateur Amazon EC2.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, choisissez la région dans laquelle votre instance externe est inscrite.
3. Dans le panneau de navigation, choisissez Clusters, puis sélectionnez le cluster qui héberge l'instance.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet.
5. Sous Constainer instances (Instances de conteneur), sélectionnez l'ID d'instance pour annuler l'enregistrement. Vous êtes redirigé vers la page de détails de l'instance de conteneur.
6. Sur le Instance de conteneur : ***id*** page, choisissez Désenregistrer.
7. Dans le message de confirmation, Annuler l'inscription.
8. Si vous n'avez plus besoin de l'instance de conteneur, résiliez l'instance Amazon EC2 sous-jacente. Pour plus d'informations, consultez Résilier une instance dans le Guide de l'utilisateur Amazon EC2.

Drainage des instances de conteneur Amazon ECS

Il peut arriver que vous deviez supprimer une instance de conteneur de votre cluster, par exemple pour effectuer des mises à jour du système ou réduire verticalement la capacité du cluster. Amazon ECS offre la possibilité de transférer une instance de conteneur vers un état `DRAINING`. C'est ce que l'on appelle le drainage des instances de conteneur. Lorsqu'une instance de conteneur est définie sur `DRAINING`, Amazon ECS bloque la planification du placement des nouvelles tâches sur l'instance de conteneur.

Comportement de drainage pour les services

Toutes les tâches faisant partie d'un service dans un état `PENDING` sont arrêtées immédiatement. S'il existe une capacité d'instance de conteneur disponible dans le cluster, le planificateur de service va démarrer les tâches de remplacement. S'il n'y a pas assez de capacité d'instance de conteneur, un message d'événement de service sera envoyé pour indiquer le problème.

Les tâches faisant partie d'un service sur l'instance de conteneur qui se trouvent à l'état `RUNNING` passent à l'état `STOPPED`. Le planificateur de service tente de remplacer les tâches en fonction du type de déploiement du service et des paramètres de configuration du déploiement, `minimumHealthyPercent` et `maximumPercent`. Pour plus d'informations, consultez Services Amazon ECS et Paramètres de définition d'un service Amazon ECS.

- Si `minimumHealthyPercent` est inférieur à 100 %, le planificateur peut ignorer `desiredCount` temporairement pendant le remplacement de tâche. Par exemple, si le `desiredCount` est de quatre tâches, un minimum de 50 % permet au planificateur d'arrêter deux tâches existantes avant de commencer deux nouvelles tâches. Si le minimum est de 100 %, le planificateur de service ne peut pas supprimer les tâches existantes tant que les tâches de remplacement sont considérées comme saines. Si les tâches des services qui n'utilisent pas d'équilibreur de charge ont l'état `RUNNING`, elles sont considérées comme saines. Les tâches des services qui utilisent un équilibreur de charge sont considérées comme saines si elles se trouvent à l'état `RUNNING` et si l'instance de conteneur sur laquelle elles sont hébergées est signalée comme saine par l'équilibreur de charge.

> **Important**
>
> Si vous utilisez des instances Spot et que la valeur `minimumHealthyPercent` est supérieure ou égale à 100 %, le service ne disposera pas de suffisamment de temps pour remplacer la tâche avant la fin de l'instance Spot.

- Le paramètre `maximumPercent` représente une limite supérieure du nombre de tâches en cours d'exécution lors du remplacement des tâches, ce qui vous permet de définir la taille du lot de remplacement. Par exemple, si le `desiredCount` est de quatre tâches, un maximum de 200 % lance quatre nouvelles tâches avant d'arrêter les quatre tâches à drainer (à condition de disposer des ressources de cluster nécessaires). Si le maximum est de 100 %, les tâches de remplacement ne peuvent pas commencer tant que les tâches de drainage sont arrêtées.

> **Important**
>
> Si `minimumHealthyPercent` et `maximumPercent` sont à 100 %, alors le service ne peut pas supprimer les tâches existantes et ne peut pas non plus démarrer les tâches de remplacement. Cela empêche le drainage des instances de conteneur ainsi que d'effectuer de nouveaux déploiements.

Comportement de drainage pour les tâches autonomes

Toutes les tâches autonomes à l'état `PENDING` ou `RUNNING` ne sont pas affectées. Vous devez attendre qu'elles s'arrêtent d'elles-mêmes ou les arrêter manuellement. L'instance de conteneur restera dans l'état `DRAINING`.

Comportement du drainage pour les instances gérées Amazon ECS

La résiliation d'Amazon ECS Managed Instances garantit des transitions de charge de travail harmonieuses tout en optimisant les coûts et en préservant l'état de santé du système. Le système de résiliation offre trois voies décisionnelles distinctes pour la résiliation d’instance, chacune présentant des caractéristiques différentes en termes de délais et d’impact sur le client.

Customer-initiated résiliation

Permet de contrôler directement la suppression des instances lorsque vous devez immédiatement retirer des instances de conteneur du service. Tu cours `deregister-container-instance`

avec le `force` paramètre de demande défini sur true. Cela signifie qu'une résiliation immédiate est requise malgré toute charge de travail en cours.

System-initiated terminaison inactive

Les instances gérées Amazon ECS surveillent en permanence et optimisent les coûts de manière proactive en mettant fin aux instances de conteneur Amazon ECS inactives qui n'exécutent aucune tâche. ECS utilise un délai heuristique pour donner aux instances de conteneurs la possibilité d'acquérir les tâches récemment lancées avant d'être interrompues. Cela peut être personnalisé avec `scaleInAfter` Paramètre de configuration du fournisseur de capacité Amazon ECS Managed Instances.

Résiliation provoquée par l'actualisation de l'infrastructure

Amazon ECS Managed Instances gère et met à jour automatiquement les logiciels des instances de conteneurs gérées afin de garantir la sécurité et la conformité tout en maintenant la disponibilité de la charge de travail. Pour plus d'informations, voir application de correctifs dans les instances gérées Amazon ECS.

Le système de résiliation met en œuvre une approche en deux phases qui équilibre la continuité de la charge de travail avec les exigences de gestion de l'infrastructure.

Phase 1 : période d'achèvement progressif

Au cours de cette phase, le système met en œuvre des stratégies de drainage progressif qui privilégient la continuité de la charge de travail. Les tâches de service sont traitées de manière progressive par le biais des processus de planification habituels d'Amazon ECS. Les tâches autonomes continuent de s'exécuter, car elles peuvent se terminer naturellement. Le système veille à ce que toutes les tâches atteignent l'état d'arrêt grâce à des processus d'achèvement naturels.

Phase 2 : application stricte des délais

Lorsque l'achèvement progressif ne permet pas d'atteindre les objectifs de résiliation dans des délais acceptables, le système met en œuvre une application stricte des délais. Le délai strict est généralement fixé à la date de début du drainage plus sept jours, ce qui laisse suffisamment de temps pour mener à bien l'opération tout en respectant les exigences opérationnelles. L'application comprend des procédures automatiques d'annulation de l'enregistrement forcée et l'arrêt immédiat de toutes les tâches restantes, quel que soit le statut d'achèvement.

Une instance de conteneur a terminé le drainage lorsque toutes les tâches qui s'exécutent sur l'instance passent à l'état `STOPPED`. L'instance de conteneur reste à l'état `DRAINING` jusqu'à ce

qu'elle soit réactivée ou supprimée. Vous pouvez vérifier l'état des tâches sur l'instance de conteneur à l'aide du ListTasks opération avec `containerInstance` paramètre pour obtenir une liste des tâches sur l'instance, suivie d'un DescribeTasks opération avec l'Amazon Resource Name (ARN) ou l'ID de chaque tâche pour vérifier l'état de la tâche.

Lorsque vous êtes prêt pour que l'instance de conteneur recommence à héberger des tâches, vous modifiez l'état de l'instance de conteneur de `DRAINING` en `ACTIVE`. Le planificateur de service Amazon ECS prend alors à nouveau en compte l'instance de conteneur pour le placement des tâches.

Procédure

Les étapes suivantes peuvent être utilisées pour définir une instance de conteneur à drainer à l'aide de la nouvelle Console de gestion AWS.

Vous pouvez également utiliser UpdateContainerInstancesState action de l'API ou update-container-instances-state commande pour changer le statut d'une instance de conteneur en `DRAINING`.

Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez un cluster qui héberge vos instances.
4. Sur le Cluster : ***name*** page, choisissez Infrastructures onglet. Puis, sous Container instances (Instances de conteneur), cochez la case correspondant à chaque instance de conteneur que vous souhaitez drainer.
5. Choisissez Actions, Drain (Drainer).

Modification du type ou de la taille de l'instance en dehors du groupe Auto Scaling

AWS vous recommande de maintenir l'immuabilité de votre infrastructure. Si vous devez modifier la taille des instances, vous pouvez soit :

- Effectuez une mise à l'échelle horizontale et ajoutez des instances supplémentaires. Ensuite, placez des tâches supplémentaires sur ces instances, ou vous

- Effectuez une mise à l'échelle verticale en lançant un nouveau larger/smaller instance, puis vidangez l'ancienne instance.

Ces deux approches permettront de minimiser l'impact sur la disponibilité des applications.

Si vous avez utilisé une autre méthode pour modifier l'instance, l'erreur suivante peut s'afficher :

```
Container instance type changes are not supported.
```

Lorsque cette erreur s'affiche, effectuez les opérations suivantes :

1. Lancez de nouvelles instances avec le type d'instance souhaité.
2. Videz vos anciens types d'instances. Pour plus d'informations, consultez the section called "Drainage des instances de conteneur".

Instances de conteneur EC2 Amazon ECS

L'agent Amazon ECS est un processus qui s'exécute sur chaque instance de conteneur enregistrée auprès de votre cluster. Il facilite la communication entre vos instances de conteneur et Amazon ECS.

> **Note**
>
> Sur les instances de conteneur Linux, le conteneur d'agents monte des répertoires de premier niveau tels que `/lib`, `/lib64` et `/proc`. Ceci est nécessaire pour les fonctionnalités d'ECS telles que les volumes Amazon EBS, le mode réseau `awsvpc`, Amazon ECS Service Connect et FireLens pour Amazon ECS.

Chaque version d'agent de conteneur Amazon ECS prend en charge un ensemble de différentes fonctions et fournit des correctifs des versions précédentes. Dans la mesure du possible, nous recommandons toujours d'utiliser la dernière version de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent de conteneur avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

L'agent de conteneur Amazon ECS contient l'image `amazon-ecs-pause`. Amazon ECS utilise cette image pour les tâches qui utilisent le mode réseau `awsvpc`.

Pour voir quelles fonctionnalités et améliorations sont incluses dans chaque version de l'agent, consultez https://github.com/aws/amazon-ecs-agent/releases.

> **Important**
>
> La version minimale de Docker pour des métriques fiables est la version Docker `v20.10.13` et versions plus récentes, incluses dans Amazon ECS-optimized AMI `20220607` et plus récents.
> La version de l'agent Amazon ECS `1.20.0` et les versions plus récentes ne prennent plus en charge les versions de Docker antérieures à `18.01.0`.

Cycle de vie

Lorsque l'agent de conteneur Amazon ECS enregistre une instance Amazon EC2 dans votre cluster, le statut de l'instance Amazon EC2 est `ACTIVE` et le statut de connexion de l'agent est `TRUE`. Cette instance de conteneur peut exécuter des demandes de tâches.

Si vous arrêtez (sans la résilier) une instance de conteneur, son statut reste `ACTIVE`, mais le statut de connexion de l'agent devient `FALSE` en quelques minutes. Toutes les tâches qui étaient en cours d'exécution sur l'instance de conteneur s'arrêtent. Si vous redémarrez l'instance de conteneur, l'agent de conteneur se connecte à nouveau avec le service Amazon ECS service et vous êtes à nouveau en mesure d'exécuter des tâches sur l'instance.

Si vous remplacez le statut d'une instance de conteneur par `DRAINING`, les nouvelles tâches ne sont pas placées sur l'instance de conteneur. Toutes les tâches de service en cours d'exécution sur l'instance de conteneur sont supprimées, si possible, afin que vous puissiez effectuer les mises à jour du système. Pour plus d'informations, consultez Drainage des instances de conteneur Amazon ECS.

Si vous annulez l'enregistrement d'une instance de conteneur ou la résiliez, le statut de l'instance de conteneur passe immédiatement à `INACTIVE` et l'instance de conteneur ne figure plus dans la liste des instances de conteneur. Cependant, vous pouvez toujours décrire l'instance de conteneur pendant une heure après la résiliation. Après un délai d'une heure, la description de l'instance n'est plus disponible.

Vous pouvez drainer les instances manuellement ou créer un hook de cycle de vie du groupe Auto Scaling pour définir l'état de l'instance sur `DRAINING`. Pour plus d'informations sur les hooks de cycle de vie Auto Scaling, consultez Hooks de cycle de vie Amazon EC2 Auto Scaling.

Prise en charge de Docker

Amazon ECS prend en charge les deux dernières versions majeures de Docker publiées sur Amazon Linux. Actuellement, cela inclut Docker 20.10.x et Docker 25.x.

La version minimale requise de Docker pour Amazon ECS se trouve dans le Fichier de spécifications de l'agent Amazon ECS sur GitHub.

Lorsque vous utilisez Amazon ECS-optimized AMI, Docker est préinstallé et configuré pour fonctionner avec l'agent de conteneur Amazon ECS. L'AMI inclut une version de Docker testée et prise en charge par Amazon ECS.

> **Note**
>
> Bien qu'Amazon ECS prenne en charge plusieurs versions de Docker, nous vous recommandons d'utiliser la version de Docker fournie avec Amazon ECS-optimized AMI pour une compatibilité et un support optimaux.

Amazon ECS-optimized AMI

Pour plus d'informations sur Amazon ECS-optimized AMI, voir Amazon ECS-optimized AMI Linux.

Informations supplémentaires

Les pages suivantes fournissent des informations supplémentaires sur les modifications :

- Journal des modifications de l'agent Amazon ECS sur GitHub
- Notes de mise à jour pour Amazon Linux 2.
- Notes de mise à jour Docker Engine dans la documentation Docker
- Documentation du pilote NVIDIA dans la documentation NVIDIA

Configuration de l'agent de conteneur Amazon ECS

S'applique à : instances EC2

L'agent de conteneur Amazon ECS prend en charge un certain nombre d'options de configuration, dont la plupart sont définies via des variables d'environnement.

Si votre instance de conteneur a été lancée avec une variante Linux d'Amazon ECS-optimized AMI, vous pouvez définir ces variables d'environnement dans `/etc/ecs/ecs.config` fichier, puis redémarrez l'agent. Vous pouvez également écrire ces variables de configuration dans vos instances de conteneur avec les données utilisateur Amazon EC2 lors du lancement. Pour plus d'informations, consultez Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données.

Si votre instance de conteneur a été lancée avec une variante Windows d'Amazon ECS-optimized AMI, vous pouvez définir ces variables d'environnement avec PowerShell SetEnvironmentVariable commande, puis redémarrez l'agent. Pour plus d'informations, consultez la section Exécuter des commandes lorsque vous lancez une instance EC2 avec des données utilisateur fournies dans le Guide de l'utilisateur Amazon EC2 et the section called "Amorçage des instances de conteneur".

Si vous démarrez manuellement l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-optimized AMI), vous pouvez utiliser ces variables d'environnement dans docker run commande que vous utilisez pour démarrer l'agent. Utilisez ces variables avec la syntaxe `--env=`*`VARIABLE_NAME`*`=`*`VARIABLE_VALUE`*. Pour les informations sensibles, telles que les informations d'authentification des référentiels privés, vous devez stocker vos variables d'environnement d'agent dans un fichier et les transmettre toutes à la fois avec l'option `--env-file` *`path_to_env_file`*. Vous pouvez utiliser les commandes suivantes pour ajouter les variables.

```
sudo systemctl stop ecs
sudo vi /etc/ecs/ecs.config
# And add the environment variables with VARIABLE_NAME=VARIABLE_VALUE format.
sudo systemctl start ecs
```

Exécution de l'agent Amazon ECS avec l'espace de noms PID de l'hôte

Par défaut, l'agent Amazon ECS s'exécute avec son propre espace de noms PID. Dans les configurations suivantes, vous pouvez configurer l'agent Amazon ECS pour qu'il s'exécute avec l'espace de noms PID de l'hôte :

- Le mode d'application SELinux est activé.
- La politique de sécurité SELinux de Docker est définie sur true.

Vous pouvez configurer ce comportement en définissant la variable d'environnement `ECS_AGENT_PID_NAMESPACE_HOST` sur `true` dans votre fichier `/etc/ecs/ecs.config`. Lorsque cette variable est activée, `ecs-init` démarrera le conteneur d'agents Amazon ECS avec

l'espace de noms PID de l'hôte (`--pid=host`), permettant à l'agent de s'amorcer correctement dans SELinux-enforcing environnements. Ces fonctionnalités sont disponibles dans les versions `1.94.0` et ultérieures de l'agent Amazon ECS.

Pour activer cette fonctionnalité, ajoutez la ligne suivante à votre fichier `/etc/ecs/ecs.config` :

```
ECS_AGENT_PID_NAMESPACE_HOST=true
```

Après avoir effectué cette modification, redémarrez l'agent Amazon ECS pour que la modification soit prise en compte :

```
sudo systemctl restart ecs
```

Les fonctionnalités suivantes ne fonctionneront pas. Le mode d'application SELinux est activé et la politique de sécurité Docker est définie sur true, même lorsque l'option `ECS_AGENT_PID_NAMESPACE_HOST=true` est définie.

- Amazon ECS Exec
- Attacher une tâche Amazon EBS
- Connexion au service
- FireLens pour Amazon ECS

Paramètres disponibles

Pour plus d'informations sur les paramètres de configuration de l'agent de conteneur Amazon ECS disponibles, consultez Agent de conteneurs Amazon ECS sur GitHub.

Stockage de la configuration d'instance de conteneur Amazon ECS dans Amazon S3

La configuration de l'agent de conteneur Amazon ECS est contrôlée avec les variables d'environnement. Variantes Linux d'Amazon ECS-optimized AMI recherche ces variables dans `/etc/ecs/ecs.config` lorsque l'agent de conteneur démarre et configurez-le en conséquence. Non-sensative variables d'environnement, telles que `ECS_CLUSTER`, peuvent être transmis à l'instance de conteneur lors du lancement via les données utilisateur Amazon EC2 et écrits dans ce fichier sans conséquence. Toutefois, d'autres informations sensibles, telles que votre AWS informations d'identification ou `ECS_ENGINE_AUTH_DATA` variable, ne doit jamais être transmise à

une instance dans les données utilisateur ou écrite dans `/etc/ecs/ecs.config` d'une manière qui leur permettrait d'apparaître dans un `.bash_history` dossier.

Stocker les informations de configuration dans un compartiment privé d'Amazon S3 et accorder un accès en lecture seule au rôle IAM de votre instance de conteneur est un moyen pratique et sécurisé d'autoriser la configuration d'une instance de conteneur au moment du lancement. Vous pouvez stocker une copie de votre fichier `ecs.config` dans un compartiment privé. Vous pouvez ensuite utiliser les données utilisateur d'Amazon EC2 pour installer AWS CLI et copiez vos informations de configuration dans `/etc/ecs/ecs.config` au lancement de l'instance.

Pour stocker un **`ecs.config`** fichier dans Amazon S3

1. Vous devez attribuer le rôle d'instance de conteneur (ecsInstanceRole) autorisations pour avoir un accès en lecture seule à Amazon S3. Vous pouvez le faire en attribuant AmazonS3ReadOnlyAccess à la `ecsInstanceRole` rôle. Pour plus d'informations sur la manière d'associer une politique à un rôle, consultez la section Modification des autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .
2. Créez un fichier `ecs.config` contenant des variables de configuration d'agent Amazon ECS valides en utilisant le format suivant. Cet exemple configure l'authentification de registre privé. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

   ```
   ECS_ENGINE_AUTH_TYPE=dockercfg
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
   ```

   > **Note**
   >
   > Pour obtenir la liste complète des variables de configuration de l'agent Amazon ECS disponibles, consultez Agent de conteneurs Amazon ECS sur GitHub.

3. Pour stocker votre fichier de configuration, créez un compartiment privé dans Amazon S3. Pour de plus amples informations, consultez la section Création d'un compartiment dans le Guide de l'utilisateur d'Amazon Simple Storage Service.
4. Chargez le fichier `ecs.config` dans votre compartiment S3. Pour plus d'informations, consultez Chargement d'objets dans le Guide de l'utilisateur Amazon Simple Storage Service.

Pour charger un **`ecs.config`** fichier depuis Amazon S3 au lancement

1. Exécutez les précédentes procédures de la présente section pour autoriser l'accès en lecture seule d'Amazon S3 à vos instances de conteneur et stocker un fichier `ecs.config` dans un compartiment S3 privé.
2. Lancez de nouvelles instances de conteneur et utilisez l'exemple de script suivant dans les données utilisateur EC2. Le script installe AWS CLI et copie votre fichier de configuration dans `/etc/ecs/ecs.config`. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

```
#!/bin/bash
yum install -y aws-cli
aws s3 cp s3://your_bucket_name/ecs.config /etc/ecs/ecs.config
```

Installation de l'agent de conteneur Amazon ECS

Si vous souhaitez enregistrer une instance Amazon EC2 auprès de votre cluster Amazon ECS et que cette instance n'utilise pas d'AMI basée sur Amazon ECS-optimized AMI, vous pouvez installer l'agent de conteneur Amazon ECS manuellement en suivant la procédure suivante. Pour ce faire, vous pouvez télécharger l'agent à partir d'un des compartiments S3 régionaux ou à partir d'Amazon Elastic Container Registry Public. Si vous téléchargez à partir d'un des compartiments S3 régionaux, vous pouvez éventuellement vérifier la validité du fichier d'agent de conteneur à l'aide de la signature PGP.

> **Note**
>
> Les unités `systemd` pour les services Amazon ECS et Docker sont soumises à une directive qui demande d'attendre que `cloud-init` ait terminé avant de démarrer les deux services. Le processus `cloud-init` n'est pas considéré comme terminé tant que les données utilisateur Amazon EC2 n'ont pas été complètement exécutées. Par conséquent, le démarrage d'Amazon ECS ou de Docker via les données utilisateur Amazon EC2 peut provoquer un blocage. Pour démarrer l'agent de conteneur à l'aide des données utilisateur Amazon EC2, vous pouvez utiliser `systemctl enable --now --no-block ecs.service`.

Installation de l'agent de conteneur Amazon ECS sur un appareil qui n'est pasAmazon Linux instance EC2

Pour installer l’agent de conteneur Amazon ECS sur une instance Amazon EC2, vous pouvez télécharger l’agent à partir d’un des compartiments Amazon S3 régionaux et l’installer.

> **Note**
>
> Lorsque vous utilisez une AMI Linux autre qu'Amazon, votre instance Amazon EC2 nécessite une prise en charge de `cgroupfs` pour le pilote `cgroup` pour que l'agent Amazon ECS prenne en charge les limites de ressources au niveau des tâches. Pour plus d'informations, voir Agent Amazon ECS activé GitHub.

Les derniers fichiers d'agent de conteneur Amazon ECS, par région, pour chaque architecture système sont répertoriés dans le tableau suivant à titre de référence.

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
us-east-2	USA Est (Ohio)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-east-1	USA Est (Virginie du Nord)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-west-1	USA Ouest (Californie du Nord)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
us-west-2	USA Ouest (Oregon)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-east-1	Asie-Pacifique (Hong Kong)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-1	Asie-Pacifique (Tokyo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-northeast-2	Asie-Pacifique (Séoul)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-south-1	Asie-Pacifique (Mumbai)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ap-southeast-1	Asie-Pacifique (Singapour)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
ap-southeast-2	Asie-Pacifique (Sydney)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
ca-central-1	Canada (Centre)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-central-1	Europe (Francfort)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-1	Europe (Irlande)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-2	Europe (Londres)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
eu-west-3	Europe (Paris)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Région	Nom de la région	Fichiers deb init Amazon ECS	Fichiers rpm init Amazon ECS
sa-east-1	Amérique du Sud (São Paulo)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 Amazon ECS init aarch64 (aarch64)
us-gov-east-1	AWS GovCloud (US-East)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)
us-gov-west-1	AWS GovCloud (US-West)	Amazon ECS init amd64 (amd64) Amazon ECS init arm64 (arm64)	Amazon ECS init x86_64 (x86_64) Amazon ECS init aarch64 (aarch64)

Pour installer l'agent de conteneur Amazon ECS sur une instance Amazon EC2 à l'aide d'un autreAmazon Linux AMI

1. Lancez une instance Amazon EC2 avec un rôle IAM qui autorise l'accès à Amazon ECS. Pour plus d’informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
2. Connectez-vous à votre instance.
3. Installez une version prise en charge de Docker sur votre instance. Pour plus d’informations, consultez Prise en charge de Docker.
4. Vérifiez votre version de Docker pour vous assurer que votre système répond à l'exigence de version minimale. Pour plus d’informations sur la prise en charge de Docker, consultez la section Instances de conteneur EC2 Amazon ECS.

   ```
   docker --version
   ```

5. Téléchargez le fichier d'agent Amazon ECS approprié pour votre système d'exploitation et votre architecture système, puis installez-le.

Pour les architectures `deb` :

```
ubuntu:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
amazon-ecs-init-latest.amd64.deb
ubuntu:~$ sudo dpkg -i amazon-ecs-init-latest.amd64.deb
```

Pour les architectures `rpm` :

```
fedora:~$ curl -O https://s3.us-west-2.amazonaws.com/amazon-ecs-agent-us-west-2/
amazon-ecs-init-latest.x86_64.rpm
fedora:~$ sudo yum localinstall -y amazon-ecs-init-latest.x86_64.rpm
```

6. Modifiez le fichier `/lib/systemd/system/ecs.service` et ajoutez la ligne suivante à la fin de la section `[Unit]`.

```
After=cloud-final.service
```

7. (Facultatif) Pour enregistrer l'instance auprès d'un cluster autre que le cluster `default`, modifiez le fichier `/etc/ecs/ecs.config` et ajoutez le contenu suivant. L'exemple suivant spécifie le cluster `MyCluster` :

```
ECS_CLUSTER=MyCluster
```

Pour plus d'informations sur ces options ainsi que d'autres options d'exécution d'agents, consultez la page Configuration de l'agent de conteneur Amazon ECS.

> **Note**
>
> Vous pouvez éventuellement stocker vos variables d'environnement d'agent dans Amazon S3. Elles pourront être téléchargées dans vos instances de conteneur lors du lancement à l'aide des données utilisateur Amazon EC2. Nous recommandons cette approche pour les informations sensibles telles que les informations d'authentification pour les référentiels privés. Pour plus d’informations, consultez Stockage de la configuration d’instance de conteneur Amazon ECS dans Amazon S3 et En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

8. Lancez le service `ecs`.

```
ubuntu:~$ sudo systemctl start ecs
```

Exécution de l'agent Amazon ECS avec le mode réseau hôte

Lorsque vous exécutez l'agent de conteneur Amazon ECS, `ecs-init` créera le conteneur d'agent de conteneur avec le mode de réseau `host`. C'est le seul mode de réseau pris en charge pour le conteneur d'agent de conteneur.

Cela vous permet de bloquer l'accès au point de terminaison de service de métadonnées de l'instance Amazon EC2 (`http://169.254.169.254`) pour les conteneurs lancés par l'agent de conteneur. Cela permet d'assurer que les conteneurs ne sont pas en mesure d'accéder aux informations d'identification du rôle IAM à partir du profil d'instance de conteneur et impose que les tâches utilisent uniquement les informations d'identification de rôle de tâche IAM. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Cela permet également à l'agent de conteneur de ne pas devoir disputer les connexions et le trafic réseau sur la passerelle `docker0`.

Paramètres de configuration du journal de l'agent de conteneur Amazon ECS

L'agent de conteneur Amazon ECS stocke les journaux sur vos instances de conteneur.

Pour l'agent de conteneur version 1.36.0 et ultérieure, les journaux sont par défaut situés sur `/var/log/ecs/ecs-agent.log` sur les instances Linux et `C:\ProgramData\Amazon\ECS\log\ecs-agent.log` sur les instances Windows.

Pour l'agent de conteneur version 1.35.0 et antérieure, les journaux sont par défaut situés sur `/var/log/ecs/ecs-agent.log.`*`timestamp`* sur les instances Linux et `C:\ProgramData\Amazon\ECS\log\ecs-agent.log.`*`timestamp`* sur les instances Windows.

Par défaut, les journaux de l'agent font l'objet d'une rotation toutes les heures et un maximum de 24 journaux sont stockés.

Voici les variables de configuration de l'agent de conteneur qui peuvent être utilisées pour modifier le comportement de journalisation de l'agent par défaut. Pour obtenir des informations détaillées sur tous les paramètres de configuration disponibles, voir Configuration de l'agent de conteneur Amazon ECS ou le Fichier README de l'agent Amazon ECS sur GitHub.

Pour l'agent de conteneur version 1.36.0 et ultérieure, voici un exemple de fichier journal lorsque le format `logfmt` est utilisé.

```
level=info time=2019-12-12T23:43:29Z msg="Loading configuration" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-agent:latest" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Image excluded from cleanup: amazon/amazon-
ecs-pause:0.1.0" module=parse.go
level=info time=2019-12-12T23:43:29Z msg="Amazon ECS agent Version: 1.36.0, Commit:
 ca640387" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Creating root ecs cgroup: /ecs"
 module=init_linux.go
level=info time=2019-12-12T23:43:29Z msg="Creating cgroup /ecs"
 module=cgroup_controller_linux.go
level=info time=2019-12-12T23:43:29Z msg="Loading state!" module=statemanager.go
level=info time=2019-12-12T23:43:29Z msg="Event stream ContainerChange start
 listening..." module=eventstream.go
level=info time=2019-12-12T23:43:29Z msg="Restored cluster 'auto-robc'" module=agent.go
level=info time=2019-12-12T23:43:29Z msg="Restored from checkpoint file. I
 am running as 'arn:aws:ecs:us-west-2:0123456789:container-instance/auto-
robc/3330a8a91d15464ea30662d5840164cd' in cluster 'auto-robc'" module=agent.go
```

Voici un exemple de fichier journal lorsque le format JSON est utilisé.

```
{"time": "2019-11-07T22:52:02Z", "level": "info", "msg": "Starting Amazon Elastic
 Container Service Agent", "module": "engine.go"}
```

Configuration des instances de conteneur Amazon ECS pour les images Docker privées

L'agent de conteneur Amazon ECS peut authentifier au moyen de registres privés, à l'aide de l'authentification de base. Lorsque vous activez l'authentification de registres privés, vous pouvez utiliser des images Docker privées dans vos définitions de tâche. Cette fonctionnalité n’est prise en charge que pour les tâches utilisant EC2.

Une autre méthode permettant d'activer les utilisations de l'authentification dans un registre privé AWS Secrets Manager pour stocker vos informations d'identification de registre privé en toute sécurité, puis les référencer dans la définition de votre conteneur. Cela permet à vos tâches d'utiliser des images de référentiels privés. Cette méthode prend en charge les tâches utilisant EC2 ou

Fargate. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

Au lancement, l'agent de conteneur Amazon ECS recherche deux variables d'environnement :

- `ECS_ENGINE_AUTH_TYPE`, qui spécifie le type des données d'authentification envoyées.
- `ECS_ENGINE_AUTH_DATA`, qui contient les informations d'authentification réelles.

Variantes Linux d'Amazon ECS-optimized L'AMI scanne le `/etc/ecs/ecs.config` fichier pour ces variables au lancement de l'instance de conteneur et à chaque démarrage du service (avec le sudo start ecs commande). AMI qui ne sont pas Amazon ECS-optimized doit stocker ces variables d'environnement dans un fichier et les transmettre avec `--env-file` *`path_to_env_file`* option pour docker run commande qui démarre l'agent de conteneur.

> **Important**
>
> Nous déconseillons d'injecter ces variables d'environnement d'authentification au moment du lancement de l'instance avec les données utilisateur Amazon EC2 ou de les transmettre avec l'option `--env` à la commande docker run. Ces méthodes ne sont pas appropriées pour les données sensibles, par exemple les informations d'authentification. Pour plus d'informations sur l'ajout en toute sécurité d'informations d'authentification à vos instances de conteneur, consultez Stockage de la configuration d'instance de conteneur Amazon ECS dans Amazon S3.

Formats d'authentification

Il existe deux formats disponibles pour une authentification de registre privé, `dockercfg` et `docker`.

Format d'authentification dockercfg

Le format `dockercfg` utilise les informations d'authentification stockées dans le fichier de configuration qui est créé lorsque vous exécutez la commande docker login. Pour créer ce fichier, exécutez docker login sur votre système local et saisissez votre nom d'utilisateur de registre, votre mot de passe et une adresse e-mail. Vous pouvez également vous connecter à une instance de conteneur et y exécuter la commande. Selon votre version de Docker, ce fichier est enregistré sous la forme `~/.dockercfg` ou `~/.docker/config.json`.

```
cat ~/.docker/config.json
```

Sortie :

```
{
  "auths": {
    "https://index.docker.io/v1/": {
      "auth": "zq212MzEXAMPLE7o6T25Dk0i"
    }
  }
}
```

> **Important**
>
> Les nouvelles versions de Docker créent un fichier de configuration, comme indiqué dans l'exemple précédent, avec un externe `auths` objet. L'agent Amazon ECS prend uniquement en charge `dockercfg` données d'authentification au format suivant, sans `auths` objet. Si l'utilitaire jq est installé, vous pouvez extraire ces données à l'aide de la commande suivante : cat ~/.docker/config.json | jq .auths

```
cat ~/.docker/config.json | jq .auths
```

Sortie :

```
{
  "https://index.docker.io/v1/": {
    "auth": "zq212MzEXAMPLE7o6T25Dk0i",
    "email": "email@example.com"
  }
}
```

Dans l'exemple précédent, les variables d'environnement suivantes doivent être ajoutées au fichier de variables d'environnement (`/etc/ecs/ecs.config` pour Amazon ECS-optimized AMI) que l'agent de conteneur Amazon ECS charge lors de l'exécution. Si vous n'utilisez pas d'Amazon ECS-optimized AMI et vous démarrez l'agent manuellement avec docker run, spécifiez le fichier de variables d'environnement avec `--env-file` *`path_to_env_file`* option lorsque vous démarrez l'agent.

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example.com"}}
```

Vous pouvez configurer plusieurs registres privés avec la syntaxe suivante :

```
ECS_ENGINE_AUTH_TYPE=dockercfg
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"auth":"zq212MzEXAMPLE7o6T25Dk0i","email":"email@example-01.com"},"repo.example-02.com":
{"auth":"fQ172MzEXAMPLEoF7225DU0j","email":"email@example-02.com"}}
```

Format d'authentification docker

Le format `docker` utilise une représentation JSON du serveur de registre que l'agent doit utiliser pour l'authentification. Il inclut également les paramètres d'authentification requis par ce registre (par exemple, nom d'utilisateur, mot de passe et adresse e-mail pour ce compte). Pour un compte Docker Hub, la représentation JSON se présente comme suit :

```
{
  "https://index.docker.io/v1/": {
    "username": "my_name",
    "password": "my_password",
    "email": "email@example.com"
  }
}
```

Dans cet exemple, les variables d'environnement suivantes doivent être ajoutées au fichier de variables d'environnement (`/etc/ecs/ecs.config` pour Amazon ECS-optimized AMI) que l'agent de conteneur Amazon ECS charge lors de l'exécution. Si vous n'utilisez pas d'Amazon ECS-optimized AMI, et vous démarrez l'agent manuellement avec docker run, spécifiez le fichier de variables d'environnement avec `--env-file` *`path_to_env_file`* option lorsque vous démarrez l'agent.

```
ECS_ENGINE_AUTH_TYPE=docker
ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
{"username":"my_name","password":"my_password","email":"email@example.com"}}
```

Vous pouvez configurer plusieurs registres privés avec la syntaxe suivante :

```
ECS_ENGINE_AUTH_TYPE=docker
```

```
ECS_ENGINE_AUTH_DATA={"repo.example-01.com":
{"username":"my_name","password":"my_password","email":"email@example-01.com"},"repo.example-02
{"username":"another_name","password":"another_password","email":"email@example-02.com"}}
```

Procédure

Utilisez la procédure suivante pour activer les registres privés pour vos instances de conteneur.

Pour activer les registres privés sur Amazon ECS-optimized AMI

1. Connectez-vous à votre instance de conteneur à l'aide de SSH.
2. Ouvrez le fichier `/etc/ecs/ecs.config` et ajoutez les valeurs `ECS_ENGINE_AUTH_TYPE` et `ECS_ENGINE_AUTH_DATA` pour votre registre et votre compte :

   ```
   sudo vi /etc/ecs/ecs.config
   ```

   Cet exemple authentifie un compte d'utilisateur Docker Hub :

   ```
   ECS_ENGINE_AUTH_TYPE=docker
   ECS_ENGINE_AUTH_DATA={"https://index.docker.io/v1/":
   {"username":"my_name","password":"my_password","email":"email@example.com"}}
   ```

3. Vérifiez si votre agent utilise la variable d'environnement `ECS_DATADIR` pour enregistrer son état :

   ```
   docker inspect ecs-agent | grep ECS_DATADIR
   ```

   Sortie :

   ```
   "ECS_DATADIR=/data",
   ```

   > **Important**
   >
   > Si la commande précédente ne renvoie pas la variable d'environnement `ECS_DATADIR`, vous devez arrêter toutes les tâches en cours d'exécution sur cette instance de conteneur avant d'arrêter l'agent. Les agents les plus récents avec la variable d'environnement `ECS_DATADIR` enregistrent leur état et peuvent être arrêtés et

démarrés sans problèmes tandis que des tâches sont en cours d'exécution. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

4. Arrêtez le service `ecs` :

```
sudo stop ecs
```

Sortie :

```
ecs stop/waiting
```

5. Redémarrez le service `ecs`.

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2 :

```
sudo systemctl restart ecs
```

- Pour l'Amazon ECS-optimized AMI Amazon Linux :

```
sudo stop ecs && sudo start ecs
```

6. (Facultatif) Vous pouvez vérifier que l'agent est en cours d'exécution et consulter des informations sur votre nouvelle instance de conteneur en interrogeant l'opération API d'introspection d'agent. Pour plus d'informations, consultez the section called "Introspection de conteneur".

```
curl http://localhost:51678/v1/metadata
```

Nettoyage automatique des tâches et des images Amazon ECS

Chaque fois qu'une tâche est placée sur une instance de conteneur, l'agent de conteneur Amazon ECS vérifie si les images référencées dans la tâche sont les images les plus récentes de la balise spécifiée dans le référentiel. Si ce n'est pas le cas, le comportement par défaut permet à l'agent d'extraire les images de leurs référentiels respectifs. Si vous modifiez fréquemment les images dans vos tâches et services, votre stockage d'instance de conteneur peut se remplir rapidement avec des images Docker que vous n'utilisez plus et que vous n'utiliserez probablement jamais plus. Par exemple, vous pouvez utiliser une intégration et un déploiement continus (CI/CD) gazoduc.

> **Note**
>
> Il est possible de personnaliser le comportement d'extraction d'image de l'agent Amazon ECS à l'aide du paramètre `ECS_IMAGE_PULL_BEHAVIOR`. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

De même, les conteneurs appartenant à des tâches arrêtées peuvent également consommer du stockage d'instance de conteneur avec des informations de journal, des volumes de données et d'autres artefacts. Ces artefacts sont utiles pour le débogage des conteneurs qui se sont arrêtés de manière inattendue, mais la plupart de ce stockage peut être libéré en toute sécurité après une période donnée.

Par défaut, l'agent de conteneur Amazon ECS élimine automatiquement les tâches arrêtées et les images Docker qui ne sont pas utilisées par des tâches de vos instances de conteneur.

> **Note**
>
> La fonction de nettoyage automatique d'image nécessite au moins la version 1.13.0 de l'agent de conteneur Amazon ECS. Pour mettre à jour votre agent avec la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Les variables de configuration d'agent suivantes sont disponibles pour ajuster votre expérience de tâches automatisées et de nettoyage d'image. Pour plus d'informations sur la façon de définir ces variables sur vos instances de conteneur, consultez Configuration de l'agent de conteneur Amazon ECS.

`ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION`

Délai d'attente par défaut avant la suppression des conteneurs pour une tâche arrêtée. Si la valeur définie est inférieure à 1 seconde, elle est ignorée. Par défaut, ce paramètre est défini sur trois heures, mais vous pouvez réduire cette période à une seconde, si votre application le nécessite.

> **Note**
>
> Le processus de nettoyage d'image ne peut pas supprimer une image tant qu'un conteneur y fait référence. Une fois les conteneurs supprimés, toutes les images non

référencées peuvent être nettoyées en fonction des paramètres de configuration du nettoyage d'image.

`ECS_DISABLE_IMAGE_CLEANUP`

Si vous définissez cette variable sur `true`, le nettoyage automatique d'image est désactivé sur votre instance de conteneur et aucune image n'est supprimée automatiquement.

`ECS_IMAGE_CLEANUP_INTERVAL`

Cette variable spécifie à quelle fréquence le processus de nettoyage d'image automatique recherche des images à supprimer. La valeur par défaut est toutes les 30 minutes, mais vous pouvez réduire ce délai à 10 minutes pour supprimer les images plus fréquemment.

`ECS_IMAGE_MINIMUM_CLEANUP_AGE`

Cette variable spécifie le délai minimal entre le moment où une image a été extraite et celui où elle peut être supprimée. Cela permet d'empêcher le nettoyage d'images tout juste extraites. La valeur par défaut est 1 heure.

`ECS_NUM_IMAGES_DELETE_PER_CYCLE`

Cette variable spécifie le nombre d'images pouvant être supprimées en un seul cycle de nettoyage. La valeur par défaut est de 5 et la valeur minimale est de 1.

Lorsque l'agent de conteneur Amazon ECS est en cours d'exécution et que le nettoyage automatique d'image n'est pas désactivé, l'agent recherche des images Docker qui ne sont pas référencées par des conteneurs en cours d'exécution ou arrêtés à une fréquence déterminée par la variable `ECS_IMAGE_CLEANUP_INTERVAL`. Si des images inutilisées sont trouvées et qu'elles sont antérieures au délai de nettoyage minimal spécifié par la variable `ECS_IMAGE_MINIMUM_CLEANUP_AGE`, l'agent supprime le nombre maximal d'images spécifiées avec la variable `ECS_NUM_IMAGES_DELETE_PER_CYCLE`. Les images référencées le moins récemment sont supprimées en premier. Une fois les images supprimées, l'agent attend jusqu'au prochain intervalle et répète le processus.

Planification de vos conteneurs sur Amazon ECS

Amazon Elastic Container Service (Amazon ECS) est un système de simultanéité optimiste à état partagé qui offre des capacités de planification flexibles pour vos charges de travail conteneurisées. Les planificateurs Amazon ECS utilisent les informations d'état de cluster fournies par l'API Amazon ECS pour prendre des décisions de placement appropriées.

Amazon ECS fournit un planificateur de service pour les tâches et les applications de longue durée. Il offre également la possibilité d’exécuter des tâches autonomes ou planifiées pour des tâches par lots ou des tâches à exécution unique. Vous pouvez spécifier les stratégies et les contraintes de placement des tâches pour exécuter les tâches qui répondent le mieux à vos besoins. Par exemple, vous pouvez spécifier si les tâches s'exécutent sur plusieurs zones de disponibilité ou dans une seule zone de disponibilité. Vous pouvez intégrer les tâches avec vos propres planificateurs personnalisés ou tiers.

Option	Quand l’utiliser	En savoir plus
Service	Le planificateur de service est adapté aux services et applications sans état de longue durée. Le planificateur de service peut aussi s'assurer que les tâches sont enregistrées sur un équilibreur de charge Elastic Load Balancing. Vous pouvez mettre à jour vos services gérés par le planificateur de service. Cela peut inclure le déploiement d'une nouvelle définition de tâche ou la modification du nombre de tâches souhaitées en cours d'exécution. Par défaut, le planificateur de service répartit les tâches entre plusieurs zones	Services Amazon ECS

Option	Quand l'utiliser	En savoir plus
	de disponibilité. Cependant , vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées.	
Tâche autonome	Une tâche autonome convient aux processus tels que les tâches par lots qui effectuent une tâche puis s'arrêtent. Par exemple, un processus peut appeler `RunTask` lorsque le travail passe en file d'attente . La tâche extrait le travail de la file d'attente, effectue le travail, puis se termine. Avec `RunTask`, vous pouvez permettre à la stratégie de placement des tâches par défaut de distribuer les tâches de façon aléatoire sur le cluster. Cela limite les risques qu'une seule instance reçoive un nombre de tâches disproportionné.	Tâches autonomes Amazon ECS

Option	Quand l'utiliser	En savoir plus
Tâches planifiées	Une tâche planifiée convient lorsque vous avez des tâches à exécuter à des intervalles définis dans votre cluster. Vous pouvez utiliser EventBridge Planificateur pour créer un calendrier. Vous pouvez exécuter des tâches pour une opération de sauvegarde ou une analyse des journaux. Le EventBridge Le planning du planificateur que vous créez peut exécuter une ou plusieurs tâches dans votre cluster à des moments précis. Votre événement programmé peut être réglé sur un intervalle spécifique (organisé tous les *N* minutes, heures ou jours). Sinon, pour une planification plus compliquée, vous pouvez utiliser une expression `cron`.	À l'aide d'Amazon EventBridge Planificateur pour planifier les tâches Amazon ECS

Options de calcul

Avec Amazon ECS, vous pouvez spécifier l'infrastructure sur laquelle vos tâches ou services s'exécutent. Vous pouvez utiliser une stratégie de fournisseur de capacité ou un type de lancement.

Pour Fargate, les fournisseurs de capacité sont Fargate et Fargate Spot. Pour EC2, le fournisseur de capacité est le groupe Auto Scaling avec les instances de conteneur enregistrées.

La stratégie des fournisseurs de capacité répartit vos tâches entre les fournisseurs de capacité associés à votre cluster.

Seuls les fournisseurs de capacité déjà associés à un cluster et disposant d'un statut `ACTIVE` ou `UPDATING` peuvent être utilisés dans une stratégie de fournisseur de capacité. Vous pouvez associer un fournisseur de capacité à un cluster lorsque vous créez un cluster.

Dans une stratégie de fournisseur de capacité, la valeur de base facultative indique le nombre minimum de tâches qui s'exécutent sur un fournisseur de capacité spécifié. Une base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité.

La valeur de poids détermine le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié. Prenez l'exemple de code suivant. Vous avez une stratégie qui contient deux fournisseurs de capacité et que les deux ont un poids de 1. Lorsque le pourcentage de base est atteint, les tâches sont réparties équitablement entre les deux fournisseurs de capacité. Dans la même logique, supposons que vous spécifiez un poids de 1 pour capacityProviderA et un poids de 4 pour capacityProviderB. Ensuite, pour chaque tâche exécutée avec capacityProviderA, il y a quatre tâches qui utilisent capacityProviderB.

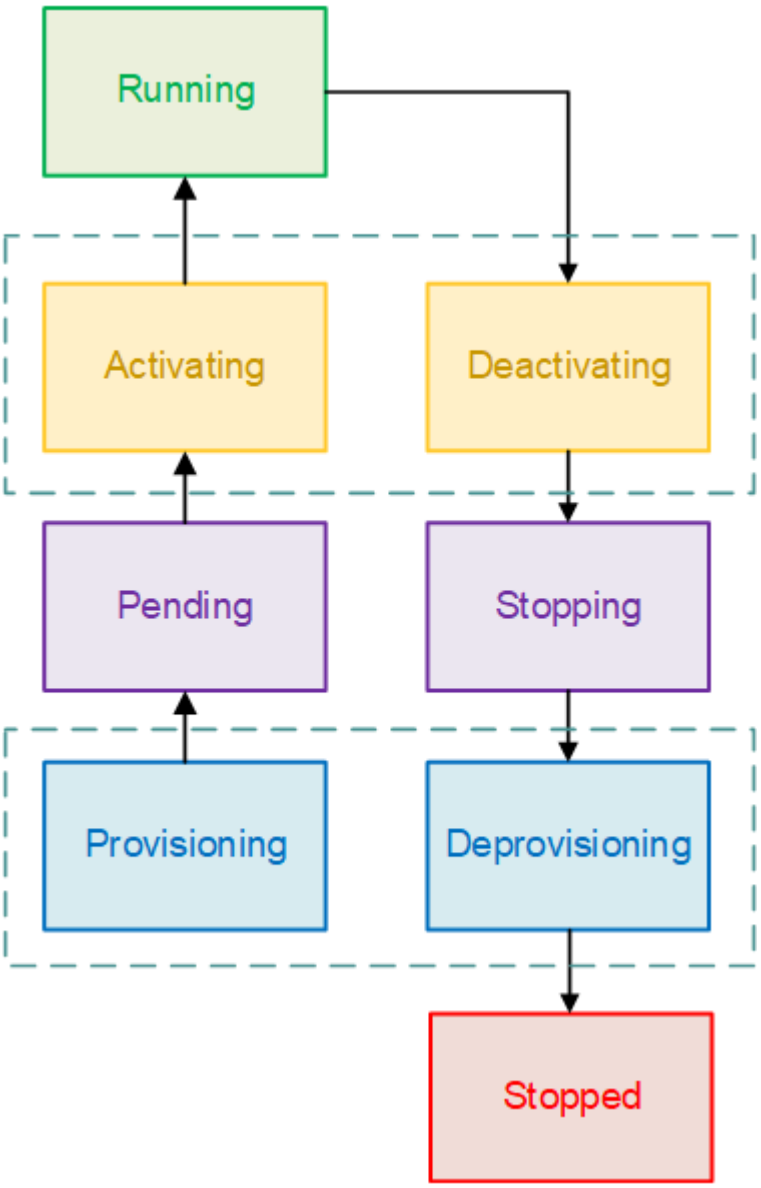
L'option de calcul lance vos tâches directement sur Fargate ou sur les instances Amazon EC2 que vous avez enregistrées manuellement dans vos clusters.

Cycle de vie des tâches Amazon ECS

Lorsqu'une tâche est lancée, soit manuellement ou dans le cadre d'un service, elle peut passer par différents états avant de se terminer d'elle-même ou d'être arrêtée manuellement. Certaines tâches sont destinées à s'exécuter en tant que tâches de traitement par lots qui passent naturellement de `PENDING` à `RUNNING` à `STOPPED`. D'autres tâches, qui peuvent être lancées dans le cadre d'un service, sont censées continuer à s'exécuter indéfiniment, ou à évoluer en fonction des besoins.

Lorsque des modifications de l'état d'une tâche sont demandées, par exemple l'arrêt d'une tâche ou la mise à jour du nombre souhaité pour un service pour le mettre à l'échelle, l'agent de conteneur Amazon ECS considère ces modifications comme le dernier état connu (`lastStatus`) de la tâche et état souhaité (`desiredStatus`) de la tâche. Le dernier état connu et l'état souhaité d'une tâche peuvent être vus dans la console ou en décrivant la tâche avec l'API ou l' AWS CLI.

Le diagramme ci-dessous illustre le flux du cycle de vie des tâches.



États de cycle de vie

Voici des descriptions de chacun des états du cycle de vie de tâche.

PROVISIONING

Amazon ECS doit effectuer des étapes supplémentaires avant que la tâche soit lancée. Par exemple, pour les tâches qui utilisent le mode de réseau `awsvpc`, l'interface réseau Elastic doit être mise en service.

PENDING

Il s'agit d'un état de transition où Amazon ECS attend l'agent de conteneur pour prendre les mesures nécessaires. Une tâche reste en attente jusqu'à ce que des ressources soient disponibles pour la tâche.

ACTIVATING

Il s'agit d'un état de transition lors duquel Amazon ECS doit effectuer des étapes supplémentaires une fois que la tâche est lancée, mais avant que la tâche passe à l'état `RUNNING`. Il s’agit de l’état dans lequel Amazon ECS extrait les images des conteneurs, crée les conteneurs, configure le réseau des tâches, enregistre les groupes cibles des équilibreurs de charge et configure la découverte de service.

RUNNING

La tâche est en cours d'exécution.

DEACTIVATING

Il s'agit d'un état de transition lors duquel Amazon ECS doit effectuer des étapes supplémentaires avant que la tâche soit arrêtée. Pour les tâches qui font partie d’un service configuré pour utiliser les groupes cibles Elastic Load Balancing, l’annulation de l’enregistrement du groupe cible se produit au cours de cet état.

STOPPING

Il s'agit d'un état de transition où Amazon ECS attend l'agent de conteneur pour prendre les mesures nécessaires.

Pour les conteneurs Linux, l'agent de conteneur enverra le signal d'arrêt défini dans l'image de votre conteneur pour indiquer que l'application doit terminer et s'arrêter à l'aide du `STOPSIGNAL` instruction. C'est `SIGTERM` par défaut. Ensuite, il enverra un `SIGKILL` après avoir attendu `StopTimeout` durée définie dans la définition de la tâche.

DEPROVISIONING

Amazon ECS doit effectuer des étapes supplémentaires une fois que la tâche est arrêtée, mais avant que la tâche passe à l'état `STOPPED`. Par exemple, pour les tâches qui utilisent le mode de réseau `awsvpc`, l'interface réseau Elastic doit être détachée et supprimée.

STOPPED

La tâche a été arrêtée avec succès.

Si votre tâche s'est arrêtée en raison d'une erreur, consultez la section Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

SUPPRIMÉ

Il s'agit d'un état de transition lorsqu'une tâche s'arrête. Cet état ne s'affiche pas sur la console, mais s'affiche sur les `describe-tasks`.

Comment Amazon ECS place les tâches sur des instances de conteneur

Vous pouvez utiliser le placement des tâches pour configurer Amazon ECS afin de placer vos tâches sur des instances de conteneur répondant à certains critères, par exemple une zone de disponibilité ou un type d'instance.

Les composants de placement des tâches sont les suivants :

- Stratégie de placement des tâches : l'algorithme permettant de sélectionner les instances de conteneur pour le placement des tâches ou les tâches à terminer. Par exemple, Amazon ECS peut sélectionner des instances de conteneur au hasard, ou il peut sélectionner des instances de conteneur de manière à ce que les tâches soient réparties uniformément entre un groupe d'instances.
- Groupe de tâches : un groupe de tâches connexes, par exemple des tâches de base de données.
- Contrainte de placement des tâches : il s'agit de règles qui doivent être respectées pour placer une tâche sur une instance de conteneur. Si la contrainte n'est pas respectée, la tâche n'est pas placée et reste dans l'état `PENDING`. Par exemple, vous pouvez utiliser une contrainte pour ne placer des tâches que sur un type d'instance spécifique.

Amazon ECS utilise différents algorithmes pour les différentes options de capacité.

Instances gérées Amazon ECS

Pour les tâches exécutées sur des instances gérées Amazon ECS, Amazon ECS doit déterminer où placer la tâche et, lors de la réduction du nombre de tâches, quelles tâches résilier. Amazon ECS prend cette décision en fonction des exigences d'instance spécifiées dans le modèle de lancement du fournisseur de capacité, des exigences spécifiées dans la définition de la tâche, telles que l'UC et la mémoire, et des contraintes de placement des tâches.

> **Note**
>
> Les instances gérées Amazon ECS ne prennent pas en charge les stratégies de placement de tâches. Amazon ECS fera tout son possible pour répartir les tâches entre les zones de disponibilité accessibles.

Lorsqu'Amazon ECS place des tâches, il a recours au processus suivant afin de sélectionner les instances de conteneur :

1. Identification des instances qui répondent aux exigences en termes d'UC, de GPU, de mémoire et de port dans la définition de tâche.
2. Identification des instances de conteneur qui satisfont aux contraintes de placement des tâches.
3. Identification des instances de conteneur qui satisfont aux exigences spécifiées dans le modèle de lancement du fournisseur de capacité.
4. Sélection des instances de conteneur pour le placement des tâches.

EC2

Pour les tâches qui utilisent le type de lancement EC2, Amazon ECS doit déterminer où placer la tâche en fonction des exigences spécifiées dans la définition de la tâche, telles que l'UC et la mémoire. De la même manière, lorsque vous réduisez le nombre de tâches, Amazon ECS doit déterminer quelles tâches doivent être résiliées. Vous pouvez appliquer des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont Amazon ECS place et résilie les tâches.

Les stratégies de placement des tâches par défaut dépendent du fait que vous exécutiez les tâches manuellement (tâches autonomes) ou dans le cadre d'un service. Pour les tâches exécutées dans le cadre d'un service Amazon ECS, la stratégie de placement des tâches est `spread`, en utilisant `attribute:ecs.availability-zone`. Il n'y a pas de contrainte de placement par défaut pour les tâches qui ne font pas partie des services. Pour plus d'informations, consultez Planification de vos conteneurs sur Amazon ECS.

> **Note**
>
> Les stratégies de placement des tâches ont une obligation de moyens, mais pas de résultat. Amazon ECS tente toujours de placer les tâches, même lorsque l'option de placement

optimale n'est pas disponible. Néanmoins, les contraintes de placement des tâches constituent une obligation et peuvent empêcher le placement des tâches.

Vous pouvez utiliser simultanément des stratégies et des contraintes de placement des tâches. Par exemple, vous pouvez utiliser une stratégie et une contrainte de placement des tâches de façon à répartir les tâches entre différentes zones de disponibilité et les regrouper par bin packing en fonction de la mémoire dans chaque zone de disponibilité, mais uniquement pour les instances G2.

Lorsqu'Amazon ECS place des tâches, il a recours au processus suivant afin de sélectionner les instances de conteneur :

1. Identification des instances qui répondent aux exigences en termes d'UC, de GPU, de mémoire et de port dans la définition de tâche.
2. Identification des instances de conteneur qui satisfont aux contraintes de placement des tâches.
3. Identification des instances de conteneur qui satisfont aux stratégies de placement des tâches.
4. Sélection des instances de conteneur pour le placement des tâches.

Fargate

Les stratégies et contraintes de placement des tâches ne sont pas prises en charge pour les tâches utilisant Fargate. Fargate fera de son mieux pour répartir les tâches entre les zones de disponibilité accessibles. Si le fournisseur de capacité inclut à la fois Fargate et Fargate Spot, le comportement de répartition est indépendant pour chaque fournisseur de capacité.

Autoscaling des instances gérées Amazon ECS et placement des tâches

Les instances gérées Amazon ECS utilisent des algorithmes intelligents pour automatiquement mettre à l'échelle la capacité de votre cluster et répartir les tâches de manière efficace sur l'ensemble de votre infrastructure. Comprendre le fonctionnement de ces algorithmes vous permet d'optimiser les configurations de vos services et de résoudre les problèmes liés aux comportements de placement.

Algorithme de placement des tâches

Les instances gérées Amazon ECS utilisent un algorithme de placement sophistiqué qui équilibre la disponibilité, l'utilisation des ressources et les exigences du réseau lors de la planification des tâches.

Répartition des zones de disponibilité

Par défaut, les instances gérées Amazon ECS donnent la priorité à la disponibilité en répartissant les tâches entre plusieurs zones de disponibilité :

- Pour les services comportant plusieurs tâches, les instances gérées Amazon ECS garantissent une distribution sur au moins trois instances dans différentes zones de disponibilité lorsque cela est possible
- Ce comportement permet une tolérance aux pannes, mais peut entraîner une baisse de l'utilisation des ressources par instance
- La répartition des zones de disponibilité prime sur l'optimisation du bin packing

Comportement de bin packing

Bien que les instances gérées Amazon ECS puissent effectuer du bin packing pour optimiser l'utilisation des ressources, ce comportement est influencé par la configuration de votre réseau :

- Pour réaliser le bin packing, configurez votre service de manière à utiliser un seul sous-réseau
- Multi-subnet les configurations privilégient la distribution des zones de disponibilité par rapport à la densité des ressources
- Le bin packing est plus probable lors du lancement initial du service que lors des événements de mise à l'échelle

Considérations relatives à la densité de l'ENI

Pour les services utilisant le mode réseau `awsvpc`, les instances gérées Amazon ECS tiennent compte de la densité de l'interface réseau Elastic (ENI) lorsqu'elles prennent des décisions de placement :

- Chaque tâche en mode `awsvpc` nécessite une ENI dédiée
- Les types d'instances ont des limites ENI différentes qui affectent la densité des tâches
- Les instances gérées Amazon ECS tiennent compte de la disponibilité de l'ENI lors de la sélection des instances cibles

> **Note**
>
> Les calculs de densité d'ENI sont continuellement améliorés afin d'optimiser les décisions de placement.

Logique de décision du fournisseur de capacité

Les fournisseurs de capacité d'instances gérées Amazon ECS prennent des décisions de mise à l'échelle et de placement en fonction de plusieurs facteurs :

Besoins en ressources

: Exigences relatives à l'UC, à la mémoire et au réseau pour les tâches en attente

Disponibilité de l'instance

: Capacité et utilisation actuelles sur les instances existantes

Contraintes de réseau

: Configuration du sous-réseau et disponibilité de l'ENI

Distribution par zone de disponibilité

: Maintien de la tolérance aux pannes dans plusieurs zones de disponibilité

Options de configuration

Stratégie de sélection de sous-réseaux

La configuration de votre sous-réseau a un impact significatif sur le comportement de placement des tâches :

Sous-réseaux multiples (par défaut)

: Priorise la répartition entre les zones de disponibilité pour une haute disponibilité

: Peut entraîner une baisse de l'utilisation des ressources par instance

: Recommandé pour les charges de production nécessitant une tolérance aux pannes

Sous-réseau unique

: Permet le bin packing pour une meilleure utilisation des ressources

Réduit la tolérance aux pannes en concentrant les tâches dans une seule zone de disponibilité

Adapté au développement ou aux charges de travail optimisées en termes de coûts

Considérations relatives au mode réseau

Le mode réseau que vous choisissez influe sur les décisions de placement :

- Mode `awsvpc` : chaque tâche nécessite une ENI dédiée, ce qui limite la densité des tâches par instance
- Mode `host` : les tâches utilisent directement le réseau de l'hôte, le placement étant principalement déterminé par la disponibilité des ressources

Considérations concernant l'architecture du processeur

Le `cpuArchitecture` que vous spécifiez dans votre définition de tâche est utilisée pour placer les tâches sur une architecture spécifique. Si vous ne spécifiez pas `cpuArchitecture`, Amazon ECS essaiera de placer les tâches sur n'importe quelle architecture de processeur disponible en fonction de la configuration du fournisseur de capacité. Vous pouvez spécifier `X86_64` ou `ARM64`.

Résolution des problèmes de placement des tâches

Modèles de placement courants

La compréhension des modèles de placement attendus permet de distinguer le comportement normal des problèmes potentiels :

Distribution répartie

Tâches réparties sur plusieurs instances avec une utilisation partielle

Comportement normal lors de l'utilisation de plusieurs sous-réseaux

Indique que la disponibilité prime sur l'efficacité des ressources

Placement concentré

Tâches multiples placées sur un nombre réduit d'instances avec un taux d'utilisation plus élevé

Prévu lors de l'utilisation d'une configuration de sous-réseau unique

Peut se produire lors du lancement initial du service

Distribution inégale

Certaines instances sont très utilisées tandis que d'autres restent sous-utilisées

Peut indiquer les limites de l'ENI ou les contraintes de ressources

Envisagez de revoir les types d'instances et la configuration réseau

Optimisation du comportement de placement

Pour optimiser le placement des tâches en fonction de vos besoins spécifiques :

1. Évaluez vos exigences de disponibilité par rapport à vos besoins d'optimisation des coûts
2. Choisissez la configuration de sous-réseau appropriée en fonction de vos priorités
3. Sélectionnez les types d'instances dotés d'une capacité ENI adaptée à votre mode réseau
4. Surveillez les modèles de placement et ajustez la configuration selon les besoins

Bonnes pratiques

- Pour les charges de travail de production : utilisez plusieurs sous-réseaux répartis dans différentes zones de disponibilité pour garantir une haute disponibilité, en acceptant le compromis en termes d'utilisation des ressources
- Pour le développement ou les tests : envisagez une configuration de sous-réseau unique pour optimiser l'utilisation des ressources et réduire les coûts
- Pour le mode **`awsvpc`** : choisissez des types d'instances dotés d'une capacité d'ENI suffisante pour éviter les contraintes de placement
- Pour optimiser les coûts : surveillez les modèles d'utilisation et ajustez la configuration des services pour équilibrer disponibilité et efficacité
- Pour résoudre les problèmes : passez en revue la configuration du sous-réseau et le mode réseau lorsque vous étudiez des modèles de placement inattendus

Utilisation des stratégies pour définir le placement des tâches Amazon ECS

Pour les tâches qui utilisent le type de lancement EC2, Amazon ECS doit déterminer où placer la tâche en fonction des exigences spécifiées dans la définition de la tâche, telles que l'UC et la mémoire. De la même manière, lorsque vous réduisez le nombre de tâches, Amazon ECS doit

déterminer quelles tâches doivent être résiliées. Vous pouvez appliquer des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont Amazon ECS place et résilie les tâches.

Les stratégies de placement des tâches par défaut dépendent du fait que vous exécutiez les tâches manuellement (tâches autonomes) ou dans le cadre d'un service. Pour les tâches exécutées dans le cadre d'un service Amazon ECS, la stratégie de placement des tâches est `spread`, en utilisant `attribute:ecs.availability-zone`. Il n'y a pas de contrainte de placement par défaut pour les tâches qui ne font pas partie des services. Pour plus d'informations, consultez Planification de vos conteneurs sur Amazon ECS.

> **Note**
>
> Les stratégies de placement des tâches ont une obligation de moyens, mais pas de résultat. Amazon ECS tente toujours de placer les tâches, même lorsque l'option de placement optimale n'est pas disponible. Néanmoins, les contraintes de placement des tâches constituent une obligation et peuvent empêcher le placement des tâches.

Vous pouvez utiliser simultanément des stratégies et des contraintes de placement des tâches. Par exemple, vous pouvez utiliser une stratégie et une contrainte de placement des tâches de façon à répartir les tâches entre différentes zones de disponibilité et les regrouper par bin packing en fonction de la mémoire dans chaque zone de disponibilité, mais uniquement pour les instances G2.

Lorsqu'Amazon ECS place des tâches, il a recours au processus suivant afin de sélectionner les instances de conteneur :

1. Identification des instances qui répondent aux exigences en termes d'UC, de GPU, de mémoire et de port dans la définition de tâche.
2. Identification des instances de conteneur qui satisfont aux contraintes de placement des tâches.
3. Identification des instances de conteneur qui satisfont aux stratégies de placement des tâches.
4. Sélection des instances de conteneur pour le placement des tâches.

Vous spécifiez les stratégies de placement des tâches dans la définition du service ou dans la définition des tâches à l'aide du paramètre `placementStrategy`.

```
"placementStrategy": [
    {
```

```
        "field": "The field to apply the placement strategy against",
        "type": "The placement strategy to use"
    }
]
```

Vous pouvez spécifier les stratégies lorsque vous exécutez une tâche (RunTask), créez un nouveau service (CreateService), ou mettre à jour un service existant (UpdateService).

Le tableau suivant décrit les types et champs disponibles.

type	Valeurs de champ valides	
`binpack` Les tâches sont placées sur les instances de conteneur afin de laisser le moins de CPU ou de mémoire inutilisé es. Cette stratégie minimise le nombre d'instances de conteneur utilisées. Lorsque cette stratégie est utilisée et qu'une action de mise à l'échelle horizontale est entreprise, Amazon ECS résilie les tâches. Elle effectue cette opération en fonction de la quantité de ressource s laissée sur l'instance de conteneur après la résiliati on de la tâche. L'instance de conteneur qui a le plus de ressources disponibles après la résiliation de la tâche voit cette tâche résiliée.	• cpu • memory	
`random`	Non utilisé	

type	Valeurs de champ valides	
Les tâches sont placées au hasard.		
`spread` Les tâches sont placées uniformément en fonction de la valeur spécifiée. Les tâches de service sont réparties en fonction des tâches du service concerné. Les tâches autonomes sont réparties en fonction des tâches du même groupe de tâches. Pour de plus amples informations sur les groupes de tâches, consultez Tâches Amazon ECS liées au groupe. Lorsque la stratégie `spread` est utilisée et qu'une action de mise à l'échelle horizontale est entreprise, Amazon ECS sélectionne les tâches à résilier qui maintiennent un équilibre entre les zones de disponibilité. Au sein d'une zone de disponibilité, les tâches sont sélectionnées au hasard.	• `instanceId` (ou `host`, ce qui a le même effet) • toute plateforme ou tout attribut personnalisé appliqué à une instance de conteneur, tel que `attribute:ecs.availability-zone`	

Les stratégies de placement des tâches peuvent également être mises à jour pour les services existants. Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

Vous pouvez créer une stratégie de placement des tâches qui utilise plusieurs stratégies en créant des tableaux de stratégies dans l'ordre d'exécution souhaité. Par exemple, si vous souhaitez répartir les tâches entre les zones de disponibilité, puis les regrouper en fonction de la mémoire au sein de chaque zone de disponibilité, spécifiez la stratégie de zone de disponibilité suivie de la stratégie de mémoire. Pour consulter des exemples de stratégies, voir Exemples de stratégies de placement des tâches Amazon ECS.

Exemples de stratégies de placement des tâches Amazon ECS

Vous pouvez définir des stratégies de placement des tâches à l'aide des actions suivantes : CreateService, UpdateService, et RunTask.

Exemples

- Répartition des tâches de manière uniforme sur plusieurs zones de disponibilité
- Répartition des tâches de manière uniforme sur toutes les instances
- Regroupement des tâches en fonction de la mémoire
- Placement des tâches de façon aléatoire
- Répartition des tâches de manière uniforme entre les zones de disponibilité, puis répartition de manière uniforme entre les instances de chaque zone de disponibilité
- Répartition des tâches de manière uniforme entre les zones de disponibilité, puis regroupement des tâches en fonction de la mémoire de chaque zone de disponibilité
- Répartition des tâches de manière uniforme entre les instances, puis regroupement des tâches en fonction de la mémoire

Répartition des tâches de manière uniforme sur plusieurs zones de disponibilité

La stratégie suivante répartit les tâches de façon uniforme dans les zones de disponibilité.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    }
]
```

Répartition des tâches de manière uniforme sur toutes les instances

La stratégie suivante répartit les tâches de façon uniforme entre toutes les instances.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    }
]
```

Regroupement des tâches en fonction de la mémoire

La stratégie suivante regroupe les tâches par bin packing en fonction de la mémoire.

```
"placementStrategy": [
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Placement des tâches de façon aléatoire

La stratégie suivante place les tâches de façon aléatoire.

```
"placementStrategy": [
    {
        "type": "random"
    }
]
```

Répartition des tâches de manière uniforme entre les zones de disponibilité, puis répartition de manière uniforme entre les instances de chaque zone de disponibilité

La stratégie suivante permet de répartir les tâches de façon égale entre les zones de disponibilité, puis de les répartir de façon égale entre les instances de chaque zone de disponibilité.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "instanceId",
        "type": "spread"
```

```
    }
]
```

Répartition des tâches de manière uniforme entre les zones de disponibilité, puis regroupement des tâches en fonction de la mémoire de chaque zone de disponibilité

La stratégie suivante permet de répartir les tâches de façon égale entre les zones de disponibilité, puis de répartir les tâches par bin packing en fonction de la mémoire de chaque zone de disponibilité.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Répartition des tâches de manière uniforme entre les instances, puis regroupement des tâches en fonction de la mémoire

La stratégie suivante répartit les tâches de manière uniforme entre toutes les instances, puis regroupe les tâches en fonction de la mémoire au sein de chaque instance.

```
"placementStrategy": [
    {
        "field": "instanceId",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Tâches Amazon ECS liées au groupe

Vous pouvez identifier un ensemble de tâches connexes et les placer dans un groupe de tâches. Toutes les tâches ayant le même nom de groupe de tâches sont considérées comme un ensemble

lorsque la stratégie de placement des tâches `spread` est utilisée. Par exemple, supposons que vous exécutiez différentes applications dans un cluster, par exemple des bases de données et des serveurs web. Afin de veiller à ce que les bases de données soient réparties de façon équilibrée dans les zones de disponibilité, ajoutez-les à un groupe de tâches nommé `databases`, puis utilisez la stratégie de placement des tâches `spread`. Pour plus d'informations, consultez Utilisation des stratégies pour définir le placement des tâches Amazon ECS.

Les groupes de tâches peuvent également être utilisés comme contrainte de placement des tâches. Lorsque vous spécifiez un groupe de tâches dans la contrainte `memberOf`, les tâches sont uniquement envoyées aux instances de conteneur qui se trouvent dans le groupe de tâches spécifié. Pour obtenir un exemple, consultez Exemple de contraintes de placement de tâche Amazon ECS.

Par défaut, les tâches autonomes utilisent le nom de famille de définition de tâche (par exemple, `family:my-task-definition`) comme nom de groupe de tâches si aucun nom de groupe de tâches personnalisé n'est spécifié. Les tâches lancées dans le cadre d'un service utilisent le nom du service comme nom de groupe de tâches et ne peuvent pas être modifiées.

Les exigences suivantes s'appliquent au groupe de tâches.

- Un nom de groupe de tâches doit être composé de 255 caractères maximum.
- Chaque tâche peut faire partie d'un seul groupe.
- Après avoir lancé une tâche, vous ne pouvez pas modifier le groupe de tâches auquel elle appartient.

Définition des instances de conteneur utilisées par Amazon ECS pour les tâches

Une contrainte de placement de tâche est une règle concernant une instance de conteneur qu'Amazon ECS utilise pour déterminer si la tâche est autorisée à s'exécuter sur l'instance. Au moins une instance de conteneur doit correspondre à la contrainte. Si aucune instance ne correspond à la contrainte, la tâche reste à l'état `PENDING`. Lorsque vous créez un nouveau service ou que vous mettez à jour un service existant, vous pouvez définir des contraintes de placement des tâches pour les tâches du service.

Vous pouvez spécifier des contraintes de placement des tâches dans la définition du service, dans la définition de la tâche ou dans la tâche à l'aide du paramètre `placementConstraint`.

```
"placementConstraints": [
```

```
    {
        "expression": "The expression that defines the task placement constraints",
        "type": "The placement constraint type to use"
    }
]
```

Le tableau suivant décrit comment utiliser les paramètres.

Constraint type (Type de contrainte)	Peut être spécifié quand	
`distinctInstance` Place chaque tâche sur une instance de conteneur différente. Amazon ECS examine le statut souhaité des tâches pour le placement des tâches. Par exemple, si le statut souhaité de la tâche existante est `STOPPED` (mais que le dernier statut ne l'est pas), une nouvelle tâche entrante peut être placée sur la même instance malgré la contrainte de placement `distinctInstance`. En conséquence, vous pouvez voir deux tâches dont le dernier statut est `RUNNING` sur la même instance. **Important** Nous recommandons aux clients qui	• Exécution d'une tâche RunTask • Création d'un nouveau service CreateService,	

Constraint type (Type de contrainte)	Peut être spécifié quand	
recherchent une isolation forte pour leurs tâches d'utiliser Fargate. Fargate exécute chaque tâche dans un environnement de virtualisation matérielle. Cela garantit que ces charges de travail conteneurisées ne partagent pas les interfaces réseau, le stockage éphémère Fargate, l'UC ou la mémoire avec d'autres tâches.		
`memberOf` Place les tâches sur des instances de conteneur qui correspondent à une expression.	• Exécution d'une tâche RunTask • Création d'un nouveau service CreateService, • Création d'une nouvelle définition de tâche RegisterTaskDefinition • Création d'une nouvelle révision d'une définition de tâche RegisterTaskDefinition • Mettre à jour un service UpdateService	

Lorsque vous utilisez le type de contrainte `memberOf`, vous pouvez créer une expression à l'aide du langage de requête de cluster qui définit les instances de conteneur dans lesquelles Amazon ECS peut placer des tâches. L'expression vous permet de regrouper vos instances de conteneur par attributs. L'expression va dans le paramètre `expression` de `placementConstraint`.

Attributs d'instance de conteneur Amazon ECS

Vous pouvez ajouter à vos instances de conteneur des métadonnées personnalisées connues sous le nom d'attributs. Chaque attribut a un nom et une valeur de chaîne facultative. Vous pouvez utiliser les attributs intégrés fournis par Amazon ECS ou définir des attributs personnalisés.

Les sections suivantes contiennent des exemples d'attributs intégrés, facultatifs et personnalisés.

Built-in attributs

Amazon ECS applique automatiquement les attributs suivants à vos instances de conteneur.

`ecs.ami-id`

ID de l'AMI utilisée pour lancer l'instance. Cet attribut peut par exemple avoir la valeur `ami-1234abcd`.

`ecs.availability-zone`

Zone de disponibilité de l'instance. Cet attribut peut par exemple avoir la valeur `us-east-1a`.

`ecs.instance-type`

Type de l'instance. Cet attribut peut par exemple avoir la valeur `g2.2xlarge`.

`ecs.os-type`

Système d'exploitation de l'instance. Les valeurs possibles pour cet attribut sont `linux` et `windows`.

`ecs.os-family`

Version du système d'exploitation de l'instance.

Pour les instances Linux, la valeur valide est `LINUX`. Pour les instances Windows, ECS définit la valeur au format `WINDOWS_SERVER_<OS_Release>_<FULL or CORE>`. Les valeurs valides sont `WINDOWS_SERVER_2022_FULL`, `WINDOWS_SERVER_2022_CORE`, `WINDOWS_SERVER_20H2_CORE`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE` et `WINDOWS_SERVER_2016_FULL`.

Ceci est important pour les conteneurs Windows et Windows containers on AWS Fargate car la version du système d'exploitation de chaque conteneur Windows doit correspondre à celle de l'hôte. Si la version Windows de l'image du conteneur est différente de celle de l'hôte, le conteneur ne démarre pas. Pour plus d'informations, consultez Compatibilité avec la version du conteneur Windows sur le site web de documentation Microsoft.

Si votre cluster exécute plusieurs versions de Windows, vous pouvez vous assurer qu'une tâche est placée sur une instance EC2 exécutée sur la même version en utilisant la contrainte de placement : `memberOf(attribute:ecs.os-family == WINDOWS_SERVER_<OS_Release>_<FULL or CORE>)`. Pour plus d'informations, consultez the section called “Récupération d'Amazon ECS-optimized Métadonnées de l'AMI Windows”.

`ecs.cpu-architecture`

Architecture du processeur de l'instance. Cet attribut peut par exemple avoir la valeur `x86_64` ou `arm64`.

`ecs.vpc-id`

VPC dans lequel l'instance a été lancée. Cet attribut peut par exemple avoir la valeur `vpc-1234abcd`.

`ecs.subnet-id`

Sous-réseau utilisé par l'instance. Cet attribut peut par exemple avoir la valeur `subnet-1234abcd`.

> **Note**
>
> Les instances gérées Amazon ECS prennent en charge le sous-ensemble d’attributs suivant :
>
> - `ecs.subnet-id`
> - `ecs.availability-zone`
> - `ecs.instance-type`
> - `ecs.cpu-architecture`

Attributs facultatifs

Amazon ECS peut ajouter les attributs suivants à vos instances de conteneur.

`ecs.awsvpc-trunk-id`

Si cet attribut existe, l'instance dispose d'une interface réseau de jonction. Pour plus d'informations, consultez Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux.

`ecs.outpost-arn`

Si cet attribut existe, il contient l'Amazon Resource Name (ARN) de l'Outpost. Pour plus d'informations, consultez the section called "Amazon Elastic Container Service sur AWS Outposts".

`ecs.capability.external`

Si cet attribut existe, l'instance est identifiée en tant qu'instance externe. Pour plus d'informations, consultez Clusters Amazon ECS pour instances externes.

Attributs personnalisés

Vous pouvez appliquer des attributs personnalisés à vos instances de conteneur. Par exemple, vous pouvez définir un attribut avec le nom « stack » et la valeur « prod ».

Lorsque vous spécifiez des attributs personnalisés, vous devez prendre en compte les éléments suivants.

- Le `name` doit comprendre entre 1 et 128 caractères et peut contenir des lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, des barres obliques et des points.
- Le `value` doit comprendre entre 1 et 128 caractères et peut contenir des lettres (majuscules et minuscules), des chiffres, des tirets, des traits de soulignement, des points, des arrobases, des barres obliques, des deux-points et des espaces. La valeur ne peut pas contenir d'espaces de début ou de fin.

Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS

Les requêtes de cluster sont des expressions qui vous permettent de regrouper des objets. Par exemple, vous pouvez regrouper des instances de conteneur par attributs, notamment par zone de disponibilité, type d'instance ou métadonnées personnalisées. Pour plus d'informations, consultez Attributs d'instance de conteneur Amazon ECS.

Une fois que vous avez défini un groupe d'instances de conteneur, vous pouvez personnaliser Amazon ECS de façon à placer les instances de conteneur en fonction du groupe. Pour plus d'informations, consultez Exécution d’une application en tant que tâche Amazon ECS et Création d’un déploiement de mise à jour propagée Amazon ECS. Vous pouvez également appliquer un filtre de groupe lorsque vous répertoriez les instances de conteneur.

Syntaxe des expressions

Les expressions ont la syntaxe suivante :

```
subject operator [argument]
```

Sujet

Attribut ou champ à évaluer.

`agentConnected`

Sélectionnez les instances de conteneur par leur état de connexion de l'agent de conteneur Amazon ECS. Vous pouvez utiliser ce filtre pour rechercher des instances avec des agents de conteneur déconnectés.

Opérateurs valides : equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`agentVersion`

Sélectionnez les instances de conteneur par la version de l'agent de conteneur Amazon ECS. Vous pouvez utiliser ce filtre pour trouver des instances qui exécutent des versions obsolètes de l'agent de conteneur Amazon ECS.

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`attribute:`*`attribute-name`*

Sélectionnez les instances de conteneur par attribut. Pour plus d’informations, consultez Attributs d’instance de conteneur Amazon ECS.

`ec2InstanceId`

Sélectionnez les instances de conteneur par leur ID d'instance Amazon EC2.

Opérateurs valides : equals (==), not_equals (!=), in, not_in (!in), matches (=~), not_matches (!~)

`registeredAt`

Sélectionnez les instances de conteneur par leur date d'enregistrement d'instance de conteneur. Vous pouvez utiliser ce filtre pour trouver de nouvelles instances enregistrées ou des instances très anciennes.

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

Formats de date valides : 2018-06-18T22:28:28+ 00:00, 18/06/2018T22:28:28Z, 18/06/2018T22:28:28, 18/06/2018

`runningTasksCount`

Sélectionnez les instances de conteneur par nombre de tâches en cours d'exécution. Vous pouvez utiliser ce filtre pour trouver des instances vides ou presque vides (peu de tâches exécutées sur elles).

Opérateurs valides : equals (==), not_equals (!=), greater_than (>), greater_than_equal (>=), less_than (<), less_than_equal (<=)

`task:group`

Sélectionnez les instances de conteneur par groupe de tâches. Pour plus d'informations, consultez Tâches Amazon ECS liées au groupe.

Opérateur

Opérateur de comparaison. Les opérateurs suivants sont pris en charge.

Opérateur	Description
==, equals	Égalité de chaînes
!=, not_equals	Inégalité de chaînes
>, greater_than	Supérieur à
>=, greater_than_equal	Supérieur ou égal à
<, less_than	Inférieur à

Opérateur	Description
<=, less_than_equal	Inférieur ou égal à
exists	Le sujet existe
!exists, not_exists	Le sujet n'existe pas
in	Valeur présente dans la liste d'arguments
!in, not_in	Valeur ne se trouvant pas dans la liste d'arguments
=~, matches	Correspondance de modèle
!~, not_matches	Non-correspondance de modèle

> **Note**
>
> Une expression ne peut pas contenir de parenthèses. En revanche, vous pouvez utiliser des parenthèses pour spécifier la priorité dans des expressions composées.

Argument

Pour de nombreux opérateurs, l'argument est une valeur littérale.

Les opérateurs `in` et `not_in` attendent une liste d'arguments comme argument. Vous spécifiez une liste d'arguments comme suit :

```
[argument1, argument2, ..., argumentN]
```

Les opérateurs matches et not_matches attendent un argument respectant la syntaxe de l'expression régulière Java. Pour en savoir plus, consultez java.util.regex.Pattern.

Expressions composées

Vous pouvez combiner des expressions à l'aide des opérateurs booléens suivants :

- &&, and
- ||, or

- !, not

Vous pouvez spécifier la priorité à l'aide de parenthèses :

```
(expression1 or expression2) and expression3
```

Exemples d'expressions

Des exemples d'expressions sont présentés ci-après.

Exemple : égalité des chaînes

L'expression suivante sélectionne des instances avec le type d'instance spécifié.

```
attribute:ecs.instance-type == t2.small
```

Exemple : liste d'arguments

L'expression suivante sélectionne des instances dans la zone de disponibilité us-east-1a ou us-east-1b.

```
attribute:ecs.availability-zone in [us-east-1a, us-east-1b]
```

Exemple : expression composée

L'expression suivante sélectionne des instances G2 qui ne se trouvent pas dans la zone de disponibilité us-east-1d.

```
attribute:ecs.instance-type =~ g2.* and attribute:ecs.availability-zone != us-east-1d
```

Exemple : affinité des tâches

L'expression suivante sélectionne des instances qui hébergent des tâches dans le groupe `service:production`.

```
task:group == service:production
```

Exemple : Tâche Anti-Affinity

L'expression suivante sélectionne des instances qui n'hébergent pas de tâches dans le groupe de base de données.

```
not(task:group == database)
```

Exemple : nombre de tâches en cours d'exécution

L'expression suivante sélectionne des instances qui n'exécutent qu'une seule tâche.

```
runningTasksCount == 1
```

Exemple : version de l'agent de conteneur Amazon ECS

L'expression suivante sélectionne des instances qui exécutent une version de l'agent de conteneur antérieure à 1.14.5.

```
agentVersion < 1.14.5
```

Exemple : Date d'enregistrement d'instance

L'expression suivante sélectionne des instances qui ont été enregistrées avant le 13 février 2018.

```
registeredAt < 2018-02-13
```

Exemple : ID d'instance Amazon EC2

L'expression suivante sélectionne des instances avec les ID d'instance Amazon EC2 suivants.

```
ec2InstanceId in ['i-abcd1234', 'i-wxyx7890']
```

Exemple de contraintes de placement de tâche Amazon ECS

Voici des exemples de contraintes de placement des tâches.

Cet exemple utilise la contrainte `memberOf` pour placer des tâches sur des instances t2. Il peut être spécifié à l'aide des actions suivantes : CreateService, UpdateService, RegisterTaskDefinition, et RunTask.

```
"placementConstraints": [
    {
        "expression": "attribute:ecs.instance-type =~ t2.*",
        "type": "memberOf"
    }
]
```

L'exemple utilise la contrainte `memberOf` pour placer des tâches de réplica sur des instances avec des tâches dans le groupe de tâches `daemon-service` du service démon, en respectant les stratégies de placement des tâches qui sont également spécifiées. Cette contrainte garantit que les tâches de service démon sont placées sur l'instance EC2 avant les tâches de service de réplica.

Remplacez `daemon-service` par le nom du service démon.

```
"placementConstraints": [
    {
        "expression": "task:group == service:daemon-service",
        "type": "memberOf"
    }
]
```

L'exemple utilise la contrainte `memberOf` pour placer des tâches sur des instances avec d'autres tâches dans le groupe de tâches `databases`, en respectant les stratégies de placement des tâches qui sont également spécifiées. Pour de plus amples informations sur les groupes de tâches, veuillez consulter Tâches Amazon ECS liées au groupe. Il peut être spécifié à l'aide des actions suivantes : CreateService, UpdateService, RegisterTaskDefinition, et RunTask.

```
"placementConstraints": [
    {
        "expression": "task:group == databases",
        "type": "memberOf"
    }
]
```

La contrainte `distinctInstance` place chaque tâche du groupe sur une instance différente. Il peut être spécifié à l'aide des actions suivantes : CreateService, UpdateService, et RunTask

Amazon ECS examine le statut souhaité des tâches pour le placement des tâches. Par exemple, si le statut souhaité de la tâche existante est `STOPPED` (mais que le dernier statut ne l'est pas), une nouvelle tâche entrante peut être placée sur la même instance malgré la contrainte de placement `distinctInstance`. En conséquence, vous pouvez voir deux tâches dont le dernier statut est `RUNNING` sur la même instance.

```
"placementConstraints": [
    {
        "type": "distinctInstance"
    }
```

```
]
```

Tâches autonomes Amazon ECS

Vous pouvez exécuter votre application en tant que tâche lorsqu'une application effectue un certain travail, puis s'arrête, par exemple un traitement par lots. Si vous souhaitez exécuter une tâche une seule fois, vous pouvez utiliser la console AWS CLI, des API ou des kits de développement logiciel.

Si vous devez exécuter votre application selon un calendrier basé sur un tarif, un calendrier cron ou un calendrier ponctuel, vous pouvez créer un calendrier à l'aide de EventBridge Planificateur.

Flux de travail des tâches

Lorsque vous lancez des tâches Amazon ECS (tâches autonomes ou par les services Amazon ECS), une tâche est créée et passe initialement à l'état `PROVISIONING`. Lorsqu'une tâche est dans l'état `PROVISIONING`, ni la tâche ni les conteneurs n'existent, car Amazon ECS a besoin de trouver la capacité de calcul pour placer la tâche.

Amazon ECS sélectionne la capacité de calcul appropriée pour votre tâche en fonction de votre type de lancement ou de la configuration de votre fournisseur de capacité. Vous pouvez utiliser des fournisseurs de capacité et des stratégies de fournisseur de capacité avec Fargate et EC2. Avec Fargate, vous n'avez pas à penser à l'allocation, à la configuration et à la mise à l'échelle de la capacité de votre cluster. Fargate s'occupe de toute la gestion de l'infrastructure pour vos tâches. Pour EC2, vous pouvez soit gérer la capacité de votre cluster en enregistrant des instances Amazon EC2 dans votre cluster, soit utiliser l'autoscaling de cluster pour simplifier la gestion de votre capacité de calcul. L'autoscaling de cluster prend en charge la mise à l'échelle dynamique de la capacité de votre cluster, afin que vous puissiez vous concentrer sur l'exécution des tâches. Amazon ECS détermine où la placer en fonction des exigences que vous spécifiez dans la définition de tâche, par exemple l'UC et la mémoire, ainsi que vos contraintes et stratégies de placement. Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

Si vous utilisez un fournisseur de capacité pour lequel la mise à l'échelle gérée est activée, les tâches qui ne peuvent pas être lancées en raison d'un manque de capacité de calcul sont placées dans l'état `PROVISIONING` plutôt que d'échouer immédiatement. Après avoir trouvé la capacité nécessaire pour placer votre tâche, Amazon ECS fournit les attachements nécessaires (par exemple, des interfaces réseau Elastic (ENI) pour les tâches en mode `awsvpc`). Il utilise l'agent de conteneur Amazon ECS pour extraire les images de vos conteneurs, puis les démarrer. Une fois l'allocation terminée et

les conteneurs concernés lancés, Amazon ECS fait passer la tâche à l'état `RUNNING`. Pour plus d'informations sur les états de tâche, consultez la section Cycle de vie des tâches Amazon ECS.

Exécution d'une application en tant que tâche Amazon ECS

Vous pouvez créer une tâche pour un processus ponctuel à l'aide de l' Console de gestion AWS.

Pour créer une tâche autonome (Console de gestion AWS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. La console Amazon ECS vous permet de créer une tâche autonome à partir de la page détaillée de votre cluster ou de la liste de révisions des définitions de tâches. Effectuez les étapes suivantes pour créer votre tâche autonome en fonction de la page de ressource que vous choisissez.

Pour démarrer un service à partir de	Étapes
Page détaillée du cluster	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. b. Sous l'onglet Tâches, choisissez Exécuter une tâche.
La page de révision de la définition des tâches	a. Sur la page Définitions de tâches, choisissez la famille de définitions de tâches pour afficher les révisions de cette famille. b. Sélectionnez la révision que vous souhaitez utiliser. c. Dans le menu Déploiement, choisissez Exécuter une tâche.

3. Pour Cluster existant, choisissez le cluster.

   Choisissez Créer un cluster pour exécuter la tâche sur un nouveau cluster

4. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Sous Configuration de calcul, choisissez votre option. Pour utiliser une stratégie de fournisseur de capacité, vous devez configurer vos fournisseurs de capacité au niveau du cluster.

   Si vous n'avez pas configuré votre cluster pour utiliser un fournisseur de capacité, utilisez plutôt un type de lancement.

   Si vous souhaitez exécuter vos charges de travail sur des instances gérées Amazon ECS, vous devez utiliser l'option de stratégie de fournisseur de capacité.

Méthode de distribution	Étapes
Stratégie de fournisseur de capacité	a. Dans Compute options (Options de calcul), sélectionnez Capacity provider strategy (Stratégie de fournisseur de capacité). b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Use custom (Utiliser une stratégie personnal

Méthode de distribution	Étapes
	isée) ou Add capacity provider strategy (Ajouter une stratégie de fournisseur de capacité) et définisse z votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Capacity provider (Fournisseur de capacité) et un Weight (Poids). **Note** Pour utiliser un fournisseur de capacité dans une stratégie, le fournisse ur de capacité doit être associé au cluster.

Méthode de distribution	Étapes
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque vous utilisez Fargate, pour Version de plateforme, spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

5. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Définition de tâche, saisissez la définition de tâche.

   > **Important**
   >
   > La console valide la sélection pour s'assurer que la famille et la révision de définition de tâche sélectionnées sont compatibles avec la configuration de calcul définie.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer.

   c. Pour Groupe de tâches, saisissez le nom du groupe de tâches.

6. Si votre définition de tâche utilise le mode réseau `awsvpc`, développez Networking (Mise en réseau). Effectuez les étapes suivantes pour spécifier une configuration personnalisée.

   a. Pour VPC, sélectionnez le VPC à utiliser.

   b. Pour Subnets (Sous-réseaux), sélectionnez un ou plusieurs sous-réseaux du VPC que le planificateur de tâches prend en compte lors du placement de vos tâches.

c. Pour Groupe de sécurité, vous pouvez choisir un groupe de sécurité existant ou en créer un nouveau. Pour utiliser un groupe de sécurité existant, sélectionnez-le et passez à l'étape suivante. Pour créer un nouveau groupe de sécurité, sélectionnez Create a new security group (Créer un nouveau groupe de sécurité). Vous devez spécifier un nom de groupe de sécurité et une description, puis ajouter une ou plusieurs règles entrantes pour le groupe de sécurité.

d. Pour Public IP (Adresse IP publique), indiquez si vous voulez attribuer automatiquement une adresse IP publique à l'interface réseau Elastic (ENI) de la tâche.

   AWS Fargate les tâches peuvent se voir attribuer une adresse IP publique lorsqu'elles sont exécutées dans un sous-réseau public afin qu'elles disposent d'un itinéraire vers Internet. Il n’est pas possible d’attribuer une adresse IP publique aux tâches EC2 à l’aide de ce champ. Pour plus d’informations, consultez les sections Options de mise en réseau des tâches Amazon ECS pour Fargate et Allocation d’une interface réseau pour une tâche Amazon ECS.

7. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

   Le nom et le type de volume sont configurés lors de la création d’une révision de définition de tâche et ne peuvent pas être modifiés lorsque vous exécutez une tâche autonome. Pour mettre à jour le nom et le type du volume, vous devez créer une révision de définition de tâche et exécuter une tâche à l’aide de cette nouvelle révision.

Pour configurer ce type de volume	Faites ceci
Amazon EBS	a. Pour Type de volume EBS, choisissez le type de volume EBS que vous souhaitez attacher à votre tâche. b. Pour Taille (Gio), saisissez une valeur valide pour la taille du volume en gibioctets (Gio). Vous pouvez

Pour configurer ce type de volume	Faites ceci	
	spécifier une taille de volume minimale de 1 Gio et maximale de 16 384 Gio. Cette valeur est obligatoire sauf si vous fournissez un ID d'instantané. c. Pour IOPS, entrez le nombre maximum de input/output opérations (IOPS) que le volume doit fournir. Cette valeur n'est configurable que pour les types de volume `io1`, `io2` et `gp3`. d. Pour Débit (MiB/s), entrez le débit que le volume doit fournir, en mébioctets par seconde (MiBps, ou MiB/s). Cette valeur n'est configurable que pour le type de volume `gp3`. e. Pour ID d'instantané, choisissez un instantané de volume Amazon EBS existant ou saisissez l'ARN d'un instantané si vous souhaitez créer un volume à partir d'un instantané. Vous pouvez également créer un volume vide en ne	

Pour configurer ce type de volume	Faites ceci	
	choisissant ni ne saisissant aucun ID d'instantané. f. Si vous spécifiez un Identifiant de l'instantané, vous pouvez spécifier Taux d'initialisation du volume (MiB/s). Entrez une valeur comprise entre 100 et 300, dans MiB/s, qui déterminera la vitesse de chargement des données à partir de l'instantané spécifié à l'aide de Identifiant de l'instantané pour la création de volumes. g. Pour Politique de résiliation, décochez la case si vous souhaitez que le volume configuré pour être attaché à la tâche soit préservé après la fin de la tâche. Par défaut, les volumes EBS rattachés à des tâches sont supprimés lorsque la tâche est résiliée. h. Pour Type de système de fichiers, choisissez le type de système de fichiers qui sera utilisé pour le stockage et la récupération des données	

Pour configurer ce type de volume	Faites ceci	
	sur le volume. Vous pouvez choisir le système d'exploitation par défaut ou un type de système de fichiers spécifique. La valeur par défaut pour Linux est `XFS`. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarre pas. i. Pour Rôle d'infrastructure, choisissez un rôle IAM doté des autorisations nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` au rôle, ou vous pouvez utiliser la politique comme guide pour créer et associer votre propre	

Pour configurer ce type de volume	Faites ceci	
	politique avec des autorisations répondant à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, consultez la section Rôle IAM d'infrastructure Amazon ECS. j. Pour Chiffrement, sélectionnez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si votre compte possède Chiffrement par défaut configuré, le volume sera chiffré avec AWS Key Management Service (AWS KMS) touche spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si tu choisis Personnalisé, vous pouvez spécifier un AWS KMS key de votre choix pour le chiffrement des volumes.	

Pour configurer ce type de volume	Faites ceci	
	Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement par défaut est configuré ou si vous créez un volume à partir d'un instantané chiffré. k. Si vous avez choisi Personnalisé pour Chiffrement, vous devez spécifier AWS KMS key que vous souhaitez utiliser. Pour la clé KMS, choisissez une AWS KMS key ou saisissez un ARN. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétrique gérée par le client, assurez-vous que vous disposez des autorisations appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. l. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume Amazon EBS soit en propageant des balises à	

Pour configurer ce type de volume	Faites ceci	
	partir de la définition de la tâche, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de tâche, choisissez Définitio n de tâche pour Propager des balises à partir de. Si vous choisissez Ne pas propager ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d'informa tions sur la création d'es volumes Amazon EBS, consultez la section Étiquetage d'un volume Amazon EBS.	

8. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Task Placement (Placement des tâches), puis choisissez parmi les options suivantes.

   Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

- Répartition équilibrée par AZ : permet de répartir les tâches entre les zones de disponibilité et les instances de conteneur de la zone de disponibilité.
- AZ Équilibré BinPack — Répartissez les tâches entre les zones de disponibilité et les instances de conteneurs dont la mémoire disponible est la plus faible.
- BinPack — Répartissez les tâches en fonction de la quantité minimale de processeur ou de mémoire disponible.
- Une tâche par hôte : permet de placer au maximum une tâche du service sur chaque instance de conteneur.
- Personnalisé : permet de définir votre propre stratégie de placement des tâches.

Si vous avez choisi Custom (Personnaliser), définissez l'algorithme de placement des tâches et les règles à prendre en compte lors du placement des tâches.

- Sous Strategy (Stratégie), pour Type et Field (Champ), choisissez l'algorithme et l'entité à utiliser pour l'algorithme.

  Vous pouvez saisir jusqu'à 5 stratégies maximum.
- Sous Contrainte, pour Type et Expression, choisissez la règle et l'attribut pour la contrainte.

  Par exemple, pour définir la contrainte permettant de placer des tâches sur des instances T2, pour Expression, saisissez attribute:ecs.instance-type =~ t2.*.

  Vous pouvez saisir jusqu'à 10 contraintes maximum.

9. (Facultatif) Pour remplacer le rôle IAM de la tâche ou le rôle d'exécution de tâches défini dans votre définition de tâche, développezTask overrides (Remplacements de tâche), puis effectuez les étapes suivantes :

   a. Pour Rôle de tâche, choisissez un rôle IAM pour cette tâche. Pour plus d’informations, consultez rôle IAM de tâche Amazon ECS.

      Seuls les rôles possédant une relation d'approbation `ecs-tasks.amazonaws.com` sont affichés. Pour savoir comment créer un rôle IAM pour vos tâches, consultez Création du rôle IAM de tâche.

   b. Pour Rôle d'exécution de tâche, choisissez un rôle d'exécution de tâche. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

10. (Facultatif) Pour remplacer les commandes du conteneur et les variables d'environnement, développez Container Overrides (Remplacements de conteneurs), puis développez le conteneur.

    - Pour envoyer une commande au conteneur autre que la commande de définition de tâche, dans Remplacement de commande, saisissez la commande Docker.
    - Pour ajouter une variable d'environnement, choisissez Add Environment Variable (Ajouter une variable d'environnement). Pour Key (Clé), saisissez le nom de votre variable d'environnement. Pour Value (Valeur), saisissez une valeur de chaîne pour la valeur d'environnement (sans guillemets doubles (" ")).

      AWS entoure les chaînes de guillemets doubles (» «) et transmet la chaîne au conteneur au format suivant :

      ```
      MY_ENV_VAR="This variable contains a string."
      ```

11. (Facultatif) Pour vous aider à identifier votre tâche, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Task definitions (Définitions de tâches).

    Ajoutez ou supprimez une balise.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.
    - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

12. Cliquez sur Créer.

À l'aide d'Amazon EventBridge Planificateur pour planifier les tâches Amazon ECS

EventBridge Scheduler est un planificateur sans serveur qui vous permet de créer, d'exécuter et de gérer des tâches à partir d'un service géré centralisé. Il fournit une fonctionnalité de planification ponctuelle et récurrente indépendamment des bus et des règles des événements. EventBridge

Le planificateur est hautement personnalisable et offre une évolutivité améliorée par rapport à EventBridge règles planifiées, avec un ensemble plus large d'opérations d'API cibles et AWS services. EventBridge Le planificateur fournit les programmes suivants que vous pouvez configurer pour vos tâches dans EventBridge Console du planificateur :

- Rate-based
- Cron-based

  Vous pouvez configurer des planifications basées sur cron dans n'importe quel fuseau horaire.
- One-time emplois du temps

  Vous pouvez configurer des planifications ponctuelles dans n'importe quel fuseau horaire.

Vous pouvez planifier votre Amazon ECS à l'aide d'Amazon EventBridge Planificateur.

Bien que vous puissiez créer une tâche planifiée dans la console Amazon ECS, actuellement EventBridge La console Scheduler fournit davantage de fonctionnalités.

Respectez les étapes suivantes avant de planifier une tâche :

1. Utilisez la console VPC pour obtenir les ID de sous-réseaux sur lesquels les tâches s'exécutent et les ID de groupe de sécurité pour les sous-réseaux. Pour plus d'informations, voir Sous-réseaux pour votre VPC, et Contrôlez le trafic vers votre AWS ressources utilisant des groupes de sécurité dans le Guide de l'utilisateur d'Amazon VPC.
2. Configurez le EventBridge Rôle d'exécution du planificateur. Pour plus d'informations, voir Configurer le rôle d'exécution dans le Amazon EventBridge Guide de l'utilisateur du planificateur.
3. Si vous souhaitez utiliser une stratégie de fournisseur de capacité pour exécuter la tâche, vous devez disposer d’un fournisseur de capacité associé au cluster.

Pour créer une planification à l'aide de la console

1. Ouvrez l'Amazon EventBridge Console Scheduler à https://console.aws.amazon.com/scheduler/home.
2. Sur la page Planifications, choisissez Créer une planification.
3. Sur la page Spécifier le détail de la planification, dans la section Nom et description de la planification, procédez comme suit :

a. Pour Nom de la planification, saisissez un nom à attribuer à votre planification. Par exemple : **`MyTestSchedule`**.

b. (Facultatif) Dans le champ Description, saisissez une description de la planification. Par exemple : **`TestSchedule`**.

c. Pour Groupe de planifications, choisissez un groupe de planifications. Si vous n'avez pas de groupe, choisissez par défaut. Pour créer un groupe de planifications, choisissez Crée votre propre planification.

   Vous utilisez des groupes de planifications pour leur ajouter des balises.

4. Choisissez vos options de planification.

Occurrence	Faites ceci...
One-time calendrier Une planification ponctuelle n'invoque un objectif qu'une seule fois à la date et à l'heure que vous indiquez.	Pour Date et heure, procédez comme suit : • Entrez une date valide au format `YYYY/MM/DD` . • Entrez un horodatage au format `hh:mm` de 24 heures. • Dans le champ Fuseau horaire, choisissez le fuseau horaire.
Planification récurrente Une planification récurrente invoque un objectif à un taux que vous spécifiez à l'aide d'une expression cron ou d'une expression rate.	a. Pour Schedule type (Planifier le type), effectuez l'une des étapes suivantes : • Pour utiliser une expression cron afin de définir le calendrier, choisissez Cron-based calendrier et entrez l'expression cron.

Occurrence	Faites ceci...
	• Pour utiliser une expression tarifaire afin de définir le calendrier, choisissez Rate-based calendrier et entrez l'expression du taux. Pour plus d'informations sur les expressions cron et rate, consultez Types d'horaires sur EventBridge Planificateur dans le Amazon EventBridge Guide de l'utilisateur du planificateur. b. Pour Fenêtre temporelle flexible, choisissez Désactivé pour désactiver cette option ou choisir l'une des fenêtres temporelles prédéfinies. Par exemple, si vous choisissez15 minutes et que vous définissez une planification récurrente pour invoquer son objectif une fois par heure, la planification s'exécute dans les 15 minutes suivant le début de chaque heure.

5. (Facultatif) Si vous avez choisi Planification récurrente à l'étape précédente, dans la section Délai, procédez comme suit :

a. Dans le champ Fuseau horaire, choisissez un fuseau horaire.

b. Pour Date et heure de début, entrez une date valide au format `YYYY/MM/DD`, puis spécifiez un horodatage au format `hh:mm` de 24 heures.

c. Pour Date et heure de fin, entrez une date valide au format `YYYY/MM/DD`, puis spécifiez un horodatage au format `hh:mm` de 24 heures.

6. Choisissez Suivant.

7. Sur la page Sélectionner une cible, procédez comme suit :

a. Choisissez Toutes les API, puis saisissez ECS dans la zone de recherche.

b. Sélectionnez Amazon ECS.

c. Dans la zone de recherche, entrez RunTask, puis choisissez RunTask.

d. Pour Cluster ECS, choisissez le cluster.

e. Pour Tâche ECS, choisissez la définition de tâche à utiliser pour la tâche.

f. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Développez Options de calcul, puis choisissez l'une des options suivantes

Options de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie : • Pour utiliser la stratégie de fournisseur de capacité par défaut, choisissez Utiliser la stratégie par défaut du cluster. • Pour utiliser une stratégie personnalisée, choisissez Utiliser une stratégie personnalisée. Ensuite, saisissez	

Options de calcul	Étapes
	les champs Base, Fournisseur de capacité et Poids. Pour EC2, le fournisseur de capacité est le groupe Auto Scaling.
Type de lancement	a. Choisissez Lancer l'application. b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. Lorsque Fargate est spécifié, pour Version de plateforme, spécifiez la version de plateforme à utiliser.

g. Pour Sous-réseaux, saisissez les ID de sous-réseaux dans lesquels exécuter la tâche.

h. Pour Groupes de sécurité, saisissez les ID des groupes de sécurité pour le sous-réseau.

i. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Contrainte de placement, puis saisissez les contraintes.

Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

j. (Facultatif) Pour vous aider à identifier vos tâches, sous Balises, configurez vos balises.

Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec les balises de définition des tâches, sélectionnez Activer les balises gérées par Amazon ECS.

8. Choisissez Suivant.

9. Sur la page Settings (Paramètres), procédez comme suit :

a. Pour activer la planification, sous État de la planification, activez Activer la planification.

b. Pour configurer une stratégie de nouvelles tentatives pour votre planification, sous Politique de nouvelle tentative et file d'attente de lettres mortes (DLQ), procédez comme suit :

   - Activez Réessayer.
   - Pour Durée de conservation maximale de l'événement, entrez le maximum heure (s) et minute (s) cette EventBridge Le planificateur doit conserver un événement non traité.
   - La durée maximale est 24 heures.
   - Pour Nombre maximum de tentatives, entrez le nombre maximum de fois EventBridge Le planificateur réessaie le calendrier si la cible renvoie une erreur.

     La valeur maximale est 185 nouvelles tentatives.

   Avec les politiques de nouvelle tentative, si un calendrier ne parvient pas à invoquer sa cible, EventBridge Le planificateur réexécute le calendrier. Si elle est configurée, vous devez définir la durée de rétention maximale et les nouvelles tentatives pour la planification.

c. Choisissez où EventBridge Le planificateur stocke les événements non diffusés.

Dead-letter file d'attente (DLQ) option	Faites ceci...	
Ne stockez pas	Sélectionnez Aucun.	
Enregistrez l'événement dans le même Compte AWS où vous créez le calendrier	a. Choisissez Sélectionnez une file d'attente Amazon SQS dans mon Compte AWS en tant que DLQ. b. Choisissez l'Amazon Resource Name (ARN) de la file d'attente Amazon SQS.	

Dead-letter file d'attente (DLQ) option	Faites ceci...	
Enregistrez l'événement dans un autre Compte AWS à partir de l'endroit où vous créez le calendrier	a. Choisissez Spécifier une file d'attente Amazon SQS dans un autre Comptes AWS en tant que DLQ. b. Entrez l'Amazon Resource Name (ARN) de la file d'attente Amazon SQS.	

d. Pour utiliser une clé gérée par le client afin de chiffrer votre entrée cible, sous Chiffrement, choisissez Personnaliser les paramètres de chiffrement (avancé).

   Si vous choisissez cette option, entrez un ARN de clé KMS existant ou choisissez Créez un AWS KMS keypour accéder à la console AWS KMS . Pour plus d'informations sur la façon dont EventBridge Le planificateur chiffre vos données au repos, voir Chiffrement au repos dans le Amazon EventBridge Guide de l'utilisateur du planificateur.

e. Pour Autorisations, choisissez Utiliser le rôle existant, puis sélectionnez le rôle.

   Avoir EventBridge Le planificateur crée un nouveau rôle d'exécution pour vous, choisissez Créer un nouveau rôle pour ce calendrier. Ensuite, saisissez un nom pour Nom du rôle. Si vous choisissez cette option, EventBridge Le planificateur associe au rôle les autorisations requises pour votre cible modélisée.

10. Choisissez Suivant.
11. Sur la page Examiner et créer une planification, examinez les détails de votre planification. Dans chaque section, choisissez Modifier pour revenir à cette étape et modifier ses détails.
12. Choisissez Créer une planification.

    Vous pouvez consulter la liste de vos planifications nouvelles et existantes sur la page Planifications. Sous la colonne État, vérifiez que votre nouvelle planification est activée.

Étapes suivantes

Vous pouvez utiliser le EventBridge Console Scheduler ou AWS CLI pour gérer le planning. Pour plus d'informations, voir Gérer un planning dans le Amazon EventBridge Guide de l'utilisateur du planificateur.

Arrêt d’une tâche Amazon ECS

Si vous n’avez plus besoin de maintenir une tâche autonome en cours d’exécution, vous pouvez arrêter la tâche. La console Amazon ECS permet d’arrêter facilement une ou plusieurs tâches.

Vous ne pouvez pas redémarrer une tâche autonome arrêtée.

Si vous souhaitez arrêter un service, consultez la section Suppression d’un service Amazon ECS à l’aide de la console.

Pour arrêter une tâche autonome (Console de gestion AWS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster pour accéder à la page de détails du cluster.
4. Sur la page de détails du cluster, choisissez l’onglet Tâches.
5. Vous pouvez filtrer les tâches par type de lancement à l’aide de la liste Filtrer par type de lancement.

Tâches à arrêter	Étapes
Une ou plusieurs	a. Sélectionnez les tâches, puis choisissez Arrêter, Arrêter sélectionné. b. Sur la Page de confirmation de l’arrêt de la tâche, choisissez Arrêter
Tous	⚠ Important Si vous choisissez d’arrêter toutes

Tâches à arrêter	Étapes
	les tâches à l'aide de la console, Amazon ECS arrête toutes les tâches autonomes et celles qui font partie d'un service. Par conséquent, nous vous recommand ons de faire preuve de prudence lorsque vous utilisez cette option. a. Choisissez Arrêter, Arrêter tout. b. Sur la Page de confirmat ion d'arrêt, saisissez Arrêter toutes les tâches et choisissez Arrêter.

Services Amazon ECS

Vous pouvez utiliser un service Amazon ECS service pour exécuter et gérer simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster Amazon ECS. Si l'une de vos tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service.

Le cas échéant, vous pouvez également exécuter votre service derrière un équilibreur de charge. Cet équilibreur de charge répartit le trafic sur les tâches associées au service.

Nous vous recommandons d'utiliser le planificateur de service pour les services et applications sans état de longue durée. Le planificateur de service garantit que la stratégie de planification que

vous spécifiez est suivie et replanifie les tâches en cas d'échec de l'une d'elles. Par exemple, si l'infrastructure sous-jacente échoue, le planificateur de services replanifie une tâche. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont le planificateur place et résilie les tâches. Si une tâche d'un service s'arrête, le planificateur lance une nouvelle tâche pour la remplacer. Ce processus se poursuit jusqu'à ce que votre service atteigne le nombre de tâches souhaité en fonction de la stratégie de planification que le service utilise.

Le planificateur de services remplace également les tâches jugées défectueuses après l'échec d'une surveillance de l'état du conteneur ou du groupe cible de l'équilibreur de charge. Ce remplacement dépend des paramètres de définition du service `maximumPercent` et `desiredCount`. Si une tâche est marquée comme défectueuse, le planificateur de services lance d'abord une tâche de remplacement. Ensuite, voici ce qui se passe.

- Si l'état de la tâche de remplacement est `HEALTHY`, le planificateur de service arrête la tâche défectueuse.
- Si l'état de santé de la tâche de remplacement est `UNHEALTHY`, le planificateur arrête soit la tâche de remplacement défectueuse, soit la tâche défectueuse existante pour que le nombre total de tâches soit égal à `desiredCount`.

Si le paramètre `maximumPercent` empêche le planificateur de démarrer d'abord une tâche de remplacement, le planificateur arrête une par une les tâches défectueuses au hasard pour libérer de la capacité, puis lance une tâche de remplacement. Le processus de démarrage et d'arrêt se poursuit jusqu'à ce que toutes les tâches défectueuses soient remplacées par des tâches saines. Une fois que toutes les tâches défectueuses ont été remplacées et que seules les tâches saines sont en cours d'exécution, si le nombre total de tâches dépasse le `desiredCount`, les tâches saines sont arrêtées au hasard jusqu'à ce que le nombre total de tâches soit égal à `desiredCount`. Pour plus d'informations sur `maximumPercent` et `desiredCount`, veuillez consulter Paramètres de définition de service (langue française non garantie).

Le planificateur de service inclut une logique qui limite la fréquence de redémarrage des tâches en cas d'échec de démarrage répété. Si une tâche est arrêtée sans avoir saisi d'état `RUNNING`, le planificateur de service commence à réduire les tentatives de lancement et envoie un message d'événement de service. Ce comportement évite d'utiliser inutilement les ressources pour des tâches qui ont échoué avant que vous puissiez résoudre le problème. Une fois le service mis à jour, le planificateur de service se comporte de nouveau normalement. Pour plus d'informations, consultez Logique de limitation d'un service Amazon ECS et Affichage des messages d'événement d'un service Amazon ECS.

Option de calcul d’infrastructure

Deux options de calcul permettent de distribuer vos tâches.

- Une capacity provider strategy (stratégie de fournisseurs de capacités) permet à Amazon ECS de distribuer vos tâches avec un ou plusieurs fournisseurs de capacité.

  Si vous souhaitez exécuter vos charges de travail sur des instances gérées Amazon ECS, vous devez utiliser l’option de stratégie de fournisseur de capacité.

  Pour la stratégie des fournisseurs de capacités, la console sélectionne une option de calcul par défaut. Voici l'ordre utilisé par la console pour sélectionner une valeur par défaut :
  - Si votre cluster a une stratégie de fournisseur de capacité définie par défaut, elle est sélectionnée.
  - Si votre cluster n’a pas de stratégie de fournisseur de capacité par défaut définie, mais que les fournisseurs de capacité Fargate ont été ajoutés au cluster, une stratégie de fournisseur de capacité personnalisée qui utilise le fournisseur de capacité `FARGATE` est sélectionnée.
  - Si votre cluster ne dispose pas d’une stratégie de fournisseur de capacité par défaut définie, mais qu’un ou plusieurs fournisseurs de capacité de groupe Auto Scaling ont été ajoutés au cluster, l’option Utiliser une stratégie personnalisée (avancée) est sélectionnée et vous devez définir manuellement la stratégie.
  - Si votre cluster n'a pas de stratégie de fournisseur de capacité par défaut définie et qu'aucun fournisseur de capacité n'est ajouté au cluster, le type de lancement Fargate est sélectionné.
- Un type de lancement permet à Amazon ECS de lancer vos tâches directement sur Fargate ou sur les instances EC2 enregistrées dans vos clusters.

  Si vous souhaitez exécuter vos charges de travail sur des instances gérées Amazon ECS, vous devez utiliser l’option de stratégie de fournisseur de capacité.

  Par défaut, le service démarre dans les sous-réseaux de votre VPC de cluster.

Mise à l'échelle automatique des services

L’autoscaling du service est la capacité à augmenter ou diminuer automatiquement le nombre souhaité de tâches dans votre service Amazon ECS. Amazon ECS utilise le service Application Auto Scaling pour fournir cette fonctionnalité.

Pour plus d’informations, consultez Mise à l’échelle automatique de votre service Amazon ECS.

Équilibrage de charge des services

Services Amazon ECS hébergés sur AWS Fargate prennent en charge les équilibreurs de charge d'application, les équilibreurs de charge réseau et les équilibreurs de charge de passerelle. Utilisez le tableau suivant pour savoir quel type d’équilibreur de charge utiliser.

Type d’équilibreur de charge	À utiliser dans les cas suivants	
Application Load Balancer	Parcours HTTP/HTTPS trafic (ou couche 7). Les équilibreurs de charge Application Load Balancer offrent plusieurs fonctions qui les rendent intéressants à utiliser avec les services Amazon ECS service : • Chaque service peut desservir le trafic provenant de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles. • Ils sont pris en charge par des tâches hébergées sur des instances Fargate ou EC2. • Les Application Load Balancers permettent aux conteneurs d'utiliser le mappage de port hôte dynamique (de sorte que plusieurs tâches du même service soient autorisées par instance de conteneur).	

Type d’équilibreur de charge	À utiliser dans les cas suivants
	• Les Application Load Balancers prennent en charge le routage basé sur le chemin d'accès et les règles de priorité (pour que plusieurs services puissent utiliser le même port d'écoute sur un même Application Load Balancer).
Network Load Balancer	Acheminer le trafic TCP ou UDP (ou couche 4).
Passerelle équilibreur de charge	Acheminer le trafic TCP ou UDP (ou couche 4). Utiliser des appareils virtuels, tels que des pare-feu, des systèmes de détection et de prévention des intrusions et des systèmes d’inspection approfondie des paquets.

Pour plus d’informations, consultez Utilisation de l’équilibrage de charge pour répartir le trafic des services Amazon ECS.

Services d’interconnexion

Si vous avez besoin d’une application pour vous connecter à d’autres applications qui s’exécutent en tant que services Amazon ECS, Amazon ECS propose les méthodes suivantes pour y parvenir sans équilibreur de charge :

- Service Connect : permet les communications de service à service avec découverte automatique à l’aide de noms abrégés et de ports standard.
- Découverte de service : la découverte de service utilise Route 53 pour créer un espace de noms pour votre service, ce qui permet de le découvrir via le DNS.

- Amazon VPC Lattice : VPC Lattice est un service de mise en réseau d'applications entièrement géré qui permet de connecter, sécuriser et surveiller vos services sur plusieurs comptes et VPC. Un coût y est associé.

Pour plus d'informations, consultez Interconnexion des services Amazon ECS.

Contrôleurs et stratégies de déploiement de service Amazon ECS

Avant de déployer votre service, déterminez les options de déploiement et les fonctionnalités utilisées par le service.

Stratégie de planification

Deux stratégies de planificateur de service sont disponibles :

- `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour plus d'informations, consultez Stratégie de planification de réplica.
- `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour plus d'informations, consultez Stratégie de planification de démon.

> **Note**
>
> Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Stratégie de planification de réplica

La stratégie de planification de réplica place et gère le nombre de tâches souhaité dans votre cluster.

Pour un service qui exécute des tâches sur Fargate, lorsque le planificateur de service lance de nouvelles tâches ou cesse l'exécution des tâches, le planificateur de service tente de maintenir un

équilibre entre les zones de disponibilité de votre service. Vous n'avez pas besoin de préciser les stratégies ou les contraintes de placement des tâches.

Lorsque vous créez un service qui exécute des tâches sur des instances EC2, vous pouvez, si vous le souhaitez, spécifier des stratégies et des contraintes de placement de tâches afin de personnaliser les décisions relatives à cette opération. Si aucune stratégie ou contrainte de placement de tâche n'est spécifiée, le planificateur de service répartit les tâches entre les zones de disponibilité par défaut. Le planificateur de service utilise la logique suivante :

- Détermine les instances de conteneur de votre cluster susceptibles de prendre en charge la définition de tâche de votre service (par exemple, avec le processeur, la mémoire, les ports et les attributs d'instances de conteneur exigés).
- Détermine les instances de conteneur qui respectent les contraintes de placement éventuellement définies pour le service.
- Lorsqu'un service de réplica dépend d'un service démon (par exemple, une tâche de routeur de journaux démon qui doit être exécutée avant que les tâches puissent utiliser la journalisation), créez une contrainte de placement des tâches qui garantit que les tâches du service démon sont placées sur l'instance EC2 avant les tâches du service de réplica. Pour plus d’informations, consultez Exemple de contraintes de placement de tâche Amazon ECS.
- Lorsqu'une stratégie de placement est définie, utilisez-la pour sélectionner une instance parmi les candidats restants.
- Lorsqu'aucune stratégie de placement n'est définie, utilisez la logique suivante pour équilibrer les tâches entre les zones de disponibilité (Availability Zones) de votre cluster :
  - Trie les instances de conteneur valides. Accorde la priorité aux instances ayant le plus petit nombre de tâches en cours d'exécution pour ce service dans leur zone de disponibilité respective. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C n'en ont pas, les instances de conteneur valides des zones B ou C sont considérées comme optimales pour le placement.
  - Place la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale basée sur les étapes précédentes. Privilégie les instances de conteneur ayant le plus petit nombre de tâches en cours d'exécution pour ce service.

Nous vous recommandons d’utiliser la fonctionnalité de rééquilibrage des services lorsque vous utilisez la stratégie `REPLICA`, car elle permet de garantir la haute disponibilité de votre service.

Stratégie de planification de démon

La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Le planificateur de service évalue les contraintes de placement des tâches pour les tâches en cours d'exécution et arrête les tâches qui ne respectent pas les contraintes de placement. Lorsque vous utilisez cette stratégie, vous n'avez pas besoin de spécifier un nombre souhaité de tâches, une stratégie de placement des tâches ou d'utiliser des politiques Service Auto Scaling.

Amazon ECS réserve des ressources de calcul d'instance de conteneur, y compris le CPU, la mémoire et l'interface réseau pour les tâches de démon. Lorsque vous lancez un service de démon sur un cluster avec d'autres services de réplica, Amazon ECS accorde la priorité à la tâche de démon. Cela signifie que la tâche démon est la première tâche à être lancée sur les instances et la dernière tâche à être arrêtée après l'arrêt de toutes les tâches réplica. Cette stratégie garantit que les ressources ne sont pas utilisées par les tâches de réplica en attente et sont disponibles pour les tâches de démon.

Le planificateur de service du démon ne place aucune tâche sur les instances qui ont un statut `DRAINING`. Si une instance de conteneur passe au statut `DRAINING`, les tâches du démon sont arrêtées. Le planificateur de service surveille également l'ajout de nouvelles instances de conteneur à votre cluster et leur ajoute les tâches du démon.

Lorsque vous spécifiez une configuration de déploiement, la valeur du paramètre `maximumPercent` doit être `100` (spécifiée sous forme de pourcentage), qui est la valeur par défaut utilisée si elle n'est pas définie. La valeur par défaut du paramètre `minimumHealthyPercent` est `0` (spécifiée sous forme de pourcentage).

Vous devez redémarrer le service lorsque vous modifiez les contraintes de placement pour le service de démon. Amazon ECS met à jour dynamiquement les ressources qui sont réservées sur les instances éligibles pour la tâche démon. Pour les instances existantes, le planificateur tente de placer la tâche sur l'instance.

Un nouveau déploiement démarre lorsqu'il existe une modification de la taille de la tâche ou de la réservation de ressource de conteneur dans la définition de tâche. Un nouveau déploiement démarre également lors de la mise à jour d'un service ou lors de la définition d'une révision différente de la définition de tâche. Amazon ECS récupère les réservations de CPU et de mémoire mises à jour pour le démon, puis bloque cette capacité pour la tâche démon.

Si les ressources sont insuffisantes pour l'un ou l'autre des cas ci-dessus, l'une des situations suivantes se produit :

- Le placement de la tâche échoue.
- UNE CloudWatch l'événement est généré.
- Amazon ECS continue d'essayer de planifier la tâche sur l'instance en attendant que les ressources soient disponibles.
- Amazon ECS libère toutes les instances réservées qui ne répondent plus aux critères de contrainte de placement et arrête les tâches démon correspondantes.

La stratégie de planification de démon peut être utilisée dans les cas suivants :

- Exécution de conteneurs d'applications
- Exécution de conteneurs de support pour les tâches de journalisation, de surveillance et de suivi

Les tâches qui utilisent Fargate ou les types de contrôleur de déploiement `CODE_DEPLOY` ou `EXTERNAL` ne prennent pas en charge la stratégie de planification de démon.

Lorsque le planificateur de service arrête d'exécuter les tâches, il tente de conserver un équilibre entre les zones de disponibilité de votre cluster. Le planificateur utilise la logique suivante :

- Si une stratégie de placement est définie, utilisez cette stratégie pour sélectionner les tâches à résilier. Par exemple, si une stratégie de répartition par zone de disponibilité est définie pour un service, une tâche est sélectionnée, laissant ainsi aux tâches restantes la meilleure répartition.
- Si aucune stratégie de placement n'est définie, maintenez l'équilibre entre les zones de disponibilité de votre cluster selon la logique suivante :
  - Triez les instances de conteneur valides. Accordez la priorité aux instances ayant le plus grand nombre de tâches en cours d'exécution pour ce service dans leur zone de disponibilité respective. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C en ont chacune deux tâches de service en cours d'exécution, les instances de conteneur des zones B ou C sont considérées comme optimales pour la mise hors service.
  - Arrêtez la tâche sur une instance de conteneur dans une zone de disponibilité optimale basée sur les étapes précédentes. Privilégier les instances de conteneur ayant le plus grand nombre de tâches en cours d'exécution pour ce service.

Contrôleurs de déploiement

Le contrôleur de déploiement est le mécanisme qui détermine la manière dont les tâches sont déployées pour votre service. Les options valides sont :

- ECS

  Lorsque vous créez un service qui utilise le contrôleur de déploiement `ECS`, vous pouvez choisir entre les stratégies de déploiement suivantes :

  - `ROLLING` : lorsque vous démarrez un service qui utilise la stratégie de déploiement Mise à jour propagée (`ROLLING`), le planificateur de service Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

    Les déploiements de mises à jour continues sont les mieux adaptés aux scénarios suivants :

    - Mises à jour progressives du service : vous devez mettre à jour votre service de manière incrémentielle sans mettre l'ensemble du service hors ligne en une seule fois.
    - Besoins en ressources limités : vous souhaitez éviter les coûts de ressources supplémentaires liés à l'exécution simultanée de deux environnements complets (comme l'exige blue/green déploiements).
    - Durée de déploiement acceptable : votre application peut tolérer un processus de déploiement plus long, car les mises à jour progressives remplacent les tâches une par une.
    - Restauration instantanée non requise : votre service peut tolérer un processus de restauration qui prend quelques minutes au lieu de quelques secondes.
    - Processus de déploiement simple : vous préférez une approche de déploiement simple, sans la complexité liée à la gestion de plusieurs environnements, groupes cibles et écouteurs.
    - Aucune exigence en matière d'équilibreur de charge, de Service Connect ou de VPC Lattice : votre service n'utilise pas ou ne nécessite pas d'équilibreur de charge d'application, d'équilibreur de charge réseau, de Service Connect ou de VPC Lattice (qui sont requis pour blue/green déploiements).
    - Applications avec état : votre application conserve un état qui rend difficile l'exécution de deux environnements parallèles.
    - Sensibilité aux coûts : vous souhaitez minimiser les coûts de déploiement en évitant d'exécuter des environnements en double pendant le déploiement.

Les mises à jour propagées constituent la stratégie de déploiement par défaut pour les services et fournissent un équilibre entre la sécurité du déploiement et l'efficacité des ressources pour de nombreux scénarios d'application courants.

- `BLUE_GREEN`: UNE blue/green stratégie de déploiement (`BLUE_GREEN`) est une méthodologie de publication qui réduit les temps d'arrêt et les risques en exécutant deux environnements de production identiques appelés bleu et vert. Avec Amazon ECS blue/green déploiements, vous pouvez valider les nouvelles révisions de service avant de diriger le trafic de production vers ces derniers. Cette approche fournit un moyen plus sûr de déployer les modifications avec la possibilité de revenir rapidement en arrière si nécessaire.

  Amazon ECS blue/green les déploiements sont les mieux adaptés aux scénarios suivants :

  - Validation des services : lorsque vous devez valider les nouvelles révisions de service avant de leur diriger le trafic de production.
  - Aucune durée d'indisponibilité : lorsque votre service nécessite des déploiements sans interruption
  - Restauration instantanée : lorsque vous avez besoin de pouvoir restaurer rapidement votre système si des problèmes sont détectés.
  - Exigence en matière d'équilibreur de charge, de Service Connect ou de VPC Lattice : lorsque votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, un Service Connect ou un VPC Lattice
- `LINEAR`: UNE linéaire stratégie de déploiement (`LINEAR`) déplace progressivement le trafic de l'environnement de production actuel vers un nouvel environnement par incréments de pourcentage égaux sur une période de temps spécifiée. Grâce aux déploiements linéaires d'Amazon ECS, vous pouvez contrôler le rythme du transfert du trafic et valider les nouvelles révisions de service en fonction de volumes croissants de trafic de production.

  Les déploiements linéaires Amazon ECS sont les mieux adaptés aux scénarios suivants :

  - Validation progressive : lorsque vous souhaitez valider progressivement votre nouvelle version de service alors que le trafic augmente
  - Surveillance des performances : lorsque vous avez besoin de temps pour surveiller les indicateurs et les performances pendant le déploiement
  - Minimisation des risques : lorsque vous souhaitez minimiser les risques en exposant la nouvelle version au trafic de production de manière incrémentielle

    - Exigence en matière d'équilibreur de charge, de Service Connect ou de VPC Lattice : lorsque votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, un Service Connect ou un VPC Lattice
- `CANARY`: UNE canari stratégie de déploiement (`CANARY`) déplace d'abord un petit pourcentage du trafic vers la nouvelle révision du service, puis déplace le trafic restant en une seule fois après une période spécifiée. Cela vous permet de tester la nouvelle version auprès d'un sous-ensemble d'utilisateurs avant le déploiement complet.

  Les déploiements Amazon ECS Canary sont les mieux adaptés aux scénarios suivants :

    - Tests de fonctionnalités : lorsque vous souhaitez tester de nouvelles fonctionnalités auprès d'un petit sous-ensemble d'utilisateurs avant le déploiement complet
    - Validation de la production : lorsque vous devez valider les performances et les fonctionnalités avec un trafic de production réel
    - Contrôle du rayon d'explosion : lorsque vous souhaitez minimiser le rayon d'explosion si des problèmes sont découverts dans la nouvelle version
    - Configuration requise en matière d'équilibreur de charge, de Service Connect ou de VPC Lattice : lorsque votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, un Service Connect ou un VPC Lattice
- Externe

  Utilisation d’un contrôleur de déploiement tiers.
- Blue/green déploiement (propulsé par AWS CodeDeploy)

  CodeDeploy installe une version mise à jour de l'application en tant que nouvel ensemble de tâches de remplacement et redirige le trafic de production de l'ensemble de tâches de l'application d'origine vers le jeu de tâches de remplacement. L'ensemble de tâches d'origine prend fin après un déploiement réussi. Utilisez ce contrôleur de déploiement pour vérifier le nouveau déploiement d’un service avant de lui envoyer un trafic de production.

Détection des échecs de déploiement d’Amazon ECS

Amazon ECS propose deux méthodes pour détecter les échecs de déploiement :

- Disjoncteur de déploiement
- CloudWatch Alarmes

Vous pouvez configurer les deux méthodes pour ramener automatiquement les déploiements ayant échoué au dernier état de fonctionnement connu.

Éléments à prendre en compte :

- Les deux méthodes ne prennent en charge que le déploiement de mises à jour continues et blue/green types de déploiement.
- Lorsqu'un service utilise des critères de réussite précoces, le disjoncteur de déploiement et CloudWatch l'annulation des alarmes s'applique jusqu'à ce qu'Amazon ECS termine le déploiement. Une fois le déploiement terminé, ils n'annulent plus le service. Pour plus d'informations, consultez Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces.
- Lorsque les deux méthodes sont utilisées, l'une ou l'autre peut provoquer un échec du déploiement.
- La méthode de restauration nécessite un déploiement antérieur à l'état TERMINÉ.
- EventBridge des événements sont générés pour les changements d'état de déploiement.

Détection des pannes par le disjoncteur de déploiement Amazon ECS

Le disjoncteur du circuit de déploiement est un mécanisme de mise à jour propagée qui détermine si les tâches atteignent un état stable. Le disjoncteur du circuit de déploiement dispose d'une option qui ramène automatiquement un déploiement ayant échoué à l'état `COMPLETED`. Vous pouvez personnaliser la façon dont le disjoncteur compte les défaillances et le seuil à partir duquel il se déclenche, afin que le comportement de restauration corresponde aux caractéristiques de démarrage de votre application et à votre tolérance aux échecs de tâches.

Lorsqu'un déploiement de service change d'état, Amazon ECS envoie un événement de changement d'état de déploiement de service à EventBridge. Cela permet de surveiller l'état de vos déploiements de services par programmation. Pour plus d'informations, consultez Événements de modification de l'état de déploiement de service Amazon ECS. Nous vous recommandons de créer et de surveiller un EventBridge règle avec un `eventName` de `SERVICE_DEPLOYMENT_FAILED` afin que vous puissiez effectuer des actions manuelles pour démarrer votre déploiement. Pour plus d'informations, voir Commencer avec EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

Lorsque le disjoncteur du circuit de déploiement détermine qu'un déploiement a échoué, il recherche le déploiement le plus récent à l'état `COMPLETED`. Il s'agit du déploiement qu'il utilise comme déploiement de restauration. Lorsque la restauration commence, le déploiement passe de `COMPLETED` à `IN_PROGRESS`. Cela signifie que le déploiement n'est pas éligible à une autre

restauration tant qu'il n'a pas atteint un état `COMPLETED`. Lorsque le disjoncteur du circuit de déploiement ne trouve aucun déploiement à l'état `COMPLETED`, il ne lance pas de nouvelles tâches et le déploiement est bloqué.

Lorsque vous créez un service, le planificateur assure le suivi des tâches qui n'ont pas pu être lancées en deux étapes.

- Étape 1 : le planificateur surveille les tâches pour voir si elles passent à l'état EN COURS D'EXÉCUTION.
  - Succès : le déploiement a une chance de passer à l'état TERMINÉ car plusieurs tâches sont passées à l'état EN COURS D'EXÉCUTION. Les critères de défaillance sont ignorés et le disjoncteur passe à l'étape 2.
  - Échec : par défaut, les tâches consécutives qui ne passent pas à l'état RUNNING sont prises en compte pour atteindre le seuil d'échec (`resetOnHealthyTask` est `true`). Quand `resetOnHealthyTask` est `false`, tous les échecs de tâches s'accumulent, que des tâches saines soient lancées ou non entre les échecs.
- Étape 2 : le déploiement entre dans cette phase lorsqu'au moins une tâche est EN COURS D'EXÉCUTION. Le disjoncteur vérifie les surveillances de l'état pour les tâches du déploiement actuel en cours d'évaluation. Les bilans de santé validés sont Elastic Load Balancing, AWS Cloud Map les bilans de santé des services et les bilans de santé des conteneurs.
  - Réussite : au moins une tâche est en cours d'exécution et les surveillances de l'état ont été validées.
  - Échec : les tâches remplacées en raison d'échecs liés aux surveillances de l'état ont atteint le seuil d'échec.

Tenez compte des points suivants lorsque vous utilisez la méthode du disjoncteur de déploiement sur un service. EventBridge génère la règle.

- La réponse `DescribeServices` donne un aperçu de l'état d'un déploiement, le `rolloutState` et `rolloutStateReason`. Lorsqu'un nouveau déploiement est démarré, il commence avec l'état `IN_PROGRESS`. Lorsque le service atteint un état stable, l'état du déploiement passe à `COMPLETED`. Si le service n'arrive pas à atteindre un état stable et que le disjoncteur de circuit est activé, le déploiement passe à l'état `FAILED`. Un déploiement dans un état `FAILED` ne lance aucune nouvelle tâche.
- Outre les événements de changement d'état de déploiement de service qu'Amazon ECS envoie pour les déploiements qui ont démarré et se sont terminés, Amazon ECS envoie également un

événement lorsqu'un déploiement avec le disjoncteur de circuit activé échoue. Ces événements fournissent des informations sur la raison pour laquelle un déploiement a échoué ou si un déploiement a été démarré en raison d'une restauration. Pour plus d'informations, consultez Événements de modification de l'état de déploiement de service Amazon ECS.

- Si un nouveau déploiement est démarré parce qu'un déploiement précédent a échoué et que la restauration a eu lieu, le champ `reason` de l'événement de changement d'état de déploiement de service indique que le déploiement a été démarré en raison d'une restauration.
- Le disjoncteur de circuit de déploiement est pris en charge uniquement pour les services Amazon ECS qui utilisent le contrôleur de déploiement de la mise à jour continue (ECS).
- Vous devez utiliser la console Amazon ECS ou AWS CLI lorsque vous utilisez le disjoncteur de déploiement avec CloudWatch option. Pour plus d'informations, consultez the section called "Créer un service à l'aide de paramètres définis" et create-service dans la Référence AWS Command Line Interface .

Les suivants `create-service` AWS CLI Un exemple montre comment créer un service Linux lorsque le disjoncteur de déploiement est utilisé avec l'option de restauration.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration "deploymentCircuitBreaker={enable=true,rollback=true}"
 \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Les suivants `create-service` AWS CLI Un exemple montre comment créer un service avec une configuration de disjoncteur de déploiement personnalisée qui utilise un nombre de défaillances fixe de 5 et un suivi des défaillances cumulées.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 10 \
```

```
    --deployment-configuration
"deploymentCircuitBreaker={enable=true,rollback=true,resetOnHealthyTask=false,thresholdConfigu
\
    --task-definition sample-fargate:1 \
    --launch-type FARGATE \
    --platform-family LINUX \
    --platform-version 1.4.0 \
    --network-configuration
"awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Exemple :

Le déploiement 1 est à l'état `COMPLETED`.

Le déploiement 2 ne pouvant pas démarrer, le disjoncteur du circuit revient au déploiement 1. Le déploiement 1 passe à l'état `IN_PROGRESS`.

Le déploiement 3 démarre et aucun déploiement n'est à l'état `COMPLETED`. Le déploiement 3 ne peut donc ni être restauré, ni lancer de tâches.

Seuil d'échec

Le disjoncteur de déploiement utilise un seuil de défaillance pour déterminer quand déplacer le déploiement vers un `FAILED` état. Vous pouvez configurer à la fois le mode de comptage des défaillances et la valeur de seuil elle-même.

Mode de comptage des défaillances

Le `resetOnHealthyTask` le réglage contrôle la façon dont le disjoncteur compte les échecs de tâches lors d'un déploiement.

`true` (par défaut)

Le nombre d'échecs est réinitialisé à 0 chaque fois qu'une tâche atteint un état normal. Seules les défaillances consécutives sont prises en compte dans le calcul du seuil. Ce mode est utile pour les applications susceptibles de rencontrer des échecs de démarrage intermittents avant de se stabiliser.

`false`

Les échecs de tâches s'accumulent tout au long du déploiement. Le nombre d'échecs ne se réinitialise jamais, même lorsque des tâches saines sont lancées entre deux échecs. Ce mode

permet une détection plus rapide lorsqu'un schéma de défaillances indique un déploiement problématique.

Configuration des seuils

Le `thresholdConfiguration` le réglage définit le moment où le disjoncteur se déclenche. Il contient un `type` qui détermine comment le seuil est calculé et `value` qui spécifie le pourcentage ou le nombre à utiliser.

`BOUNDED_PERCENT` (par défaut)

Amazon ECS se multiplie `value` par le dernier décompte souhaité par le service pour calculer le seuil de défaillance. Le résultat est limité à un minimum de 3 et à un maximum de 200. Il s'agit du type par défaut avec une valeur par défaut de 50.

`UNBOUNDED_PERCENT`

Amazon ECS se multiplie `value` par le dernier décompte souhaité par le service pour calculer le seuil de défaillance. Il n'y a pas de limite minimale ou maximale pour le résultat. Utilisez ce type pour les services dont le nombre souhaité est élevé et qui nécessitent un seuil proportionnel sans le plafond de 200.

`COUNT`

Amazon ECS utilise `value` directement en tant que seuil de défaillance. Le seuil reste fixe quel que soit le nombre de services souhaité. Utilisez ce type lorsque vous souhaitez obtenir un nombre exact de défaillances tolérées, par exemple un seuil inférieur pour des annulations plus rapides dans les environnements de développement.

Pour les types de pourcentages (`BOUNDED_PERCENT` et `UNBOUNDED_PERCENT`), la plage valide pour `value` est compris entre 1 et 100. Lors d'un déploiement, Amazon ECS utilise en permanence le dernier nombre de services souhaités dans ses calculs.

Comment **`POURCENT_LIMITE`** calcule le seuil

Lorsque vous utilisez la valeur par défaut `BOUNDED_PERCENT` type de seuil, le disjoncteur de déploiement calcule le seuil à l'aide de la formule suivante.

```
Minimum threshold (3) <= (value/100) * desired task count => Maximum threshold (200)
```

Lorsque le résultat du calcul est inférieur à 3, le seuil est fixé à 3. Lorsque le résultat est supérieur à 200, le seuil est fixé à 200. Dans le cas contraire, le seuil est défini sur la valeur calculée (arrondie au chiffre supérieur).

Le tableau suivant fournit des exemples utilisant la valeur par défaut de 50.

Nombre souhaité	Calculs	Threshold
1	`3 <= 0.5 * 1 => 200`	3 (la valeur calculée est inférieure au minimum)
25	`3 <= 0.5 * 25 => 200`	13 (la valeur est arrondie à la valeur supérieure)
400	`3 <= 0.5 * 400 => 200`	200 (la valeur calculée est supérieure au maximum)
800	`3 <= 0.5 * 800 => 200`	200 (la valeur calculée est supérieure au maximum)

Avec `UNBOUNDED_PERCENT`, le même calcul s'applique mais sans les limites minimale et maximale. Par exemple, un service dont le nombre souhaité est de 800 et une valeur de 50 aurait un seuil de 400.

La surveillance de l'état du déploiement comporte deux étapes.

1. Le disjoncteur de déploiement surveille les tâches qui font partie du déploiement et vérifie les tâches qui se trouvent dans l'état `RUNNING`. Le planificateur ignore les critères d'échec lorsqu'une tâche du déploiement actuel se trouve dans l'état `RUNNING` et passe à l'étape suivante. Lorsque les tâches n'atteignent pas `RUNNING` état, le disjoncteur de déploiement augmente le nombre de défaillances d'une unité. Lorsque le nombre d'échecs est égal au seuil, le déploiement est marqué comme étant `FAILED`.
2. Cette étape débute lorsqu'il y a une ou plusieurs tâches dans l'état `RUNNING`. Le disjoncteur de déploiement effectue une surveillance de l'état sur les ressources suivantes pour les tâches du déploiement actuel :
   - Equilibreurs de charge Elastic Load Balancing
   - AWS Cloud Map service

- Surveillance de l'état des conteneurs Amazon ECS

  Lorsqu'une surveillance de l'état échoue pour la tâche, le disjoncteur de déploiement augmente d'un le nombre d'échecs. Lorsque le nombre d'échecs est égal au seuil, le déploiement est marqué comme étant `FAILED`.

Par exemple, lorsque le seuil est de 3, le disjoncteur se déclenche lorsque le nombre de défaillances atteint 0. Lorsqu'une tâche ne parvient pas à atteindre l'état `RUNNING`, le disjoncteur de déploiement augmente le nombre d'échecs d'une unité. Lorsque le nombre d'échecs est égal à 3, le déploiement est marqué comme `FAILED`.

Pour obtenir des exemples supplémentaires sur l'utilisation de l'option de restauration, consultez Announcing Amazon ECS deployment circuit breaker (Annonce du disjoncteur de circuit de déploiement Amazon ECS).

Comment CloudWatch des alarmes détectent les échecs de déploiement d'Amazon ECS

Vous pouvez configurer Amazon ECS pour définir l'échec du déploiement lorsqu'il détecte qu'un CloudWatch l'alarme est entrée dans `ALARM` état.

Vous pouvez éventuellement définir la configuration pour qu'elle revienne au dernier déploiement réussi en cas d'échec d'un déploiement.

Les suivants `create-service` AWS CLI Un exemple montre comment créer un service Linux lorsque les alarmes de déploiement sont utilisées avec l'option de restauration.

```
aws ecs create-service \
     --service-name MyService \
     --deployment-controller type=ECS \
     --desired-count 3 \
     --deployment-configuration
 "alarms={alarmNames=[alarm1Name,alarm2Name],enable=true,rollback=true}" \
     --task-definition sample-fargate:1 \
     --launch-type FARGATE \
     --platform-family LINUX \
     --platform-version 1.4.0 \
     --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
```

Tenez compte des points suivants lorsque vous utilisez Amazon CloudWatch méthode d'alarme sur un service.

- La durée pendant laquelle les révisions de service bleues et vertes se déroulent simultanément après le transfert du trafic de production. Amazon ECS calcule cette période en fonction de la configuration d'alarme associée au déploiement. Vous ne pouvez pas définir cette valeur.
- Le paramètre de demande `deploymentConfiguration` contient désormais le type de données `alarms`. Vous pouvez spécifier les noms des alarmes, si vous souhaitez utiliser la méthode et si vous souhaitez lancer une restauration lorsque les alarmes indiquent un échec de déploiement. Pour plus d'informations, voir CreateService dans le Référence de l'API Amazon Elastic Container Service.
- La réponse `DescribeServices` donne un aperçu de l'état d'un déploiement, le `rolloutState` et `rolloutStateReason`. Lorsqu'un nouveau déploiement démarre, il commence à l'état `IN_PROGRESS`. Lorsque le service atteint un état stable et que la durée de l'intégration est terminée, l'état du déploiement passe à `COMPLETED`. Si le service n'arrive pas à atteindre un état stable et que l'alarme passe à l'état `ALARM`, le déploiement passe à un état `FAILED`. Un déploiement dans un état `FAILED` ne lancera aucune nouvelle tâche.
- Outre les événements de changement d'état de déploiement de service qu'Amazon ECS envoie pour les déploiements qui ont démarré et se sont terminés, Amazon ECS envoie également un événement lorsqu'un déploiement qui utilise des alarmes échoue. Ces événements fournissent des informations sur la raison pour laquelle un déploiement a échoué ou si un déploiement a été démarré en raison d'une restauration. Pour plus d’informations, consultez Événements de modification de l’état de déploiement de service Amazon ECS.
- Si un nouveau déploiement démarre parce qu'un déploiement précédent a échoué et que la restauration a été activée, le champ `reason` de l'événement de changement d'état de déploiement de service indique que le déploiement a démarré en raison d'une restauration.
- Si vous utilisez le disjoncteur de déploiement et l'Amazon CloudWatch alarmes pour détecter les pannes, l'une ou l'autre peut déclencher un échec de déploiement dès que les critères de l'une ou l'autre des méthodes sont remplis. Une restauration se produit lorsque vous utilisez l'option de restauration pour la méthode à l'origine de l'échec du déploiement.
- L'Amazonie CloudWatch les alarmes ne sont prises en charge que pour les services Amazon ECS qui utilisent la mise à jour continue (`ECS`) contrôleur de déploiement.
- Vous pouvez configurer cette option à l’aide de la console Amazon ECS ou de l’ AWS CLI. Pour plus d'informations, consultez the section called “Créer un service à l'aide de paramètres définis” et create-service dans la Référence AWS Command Line Interface .
- Vous remarquerez peut-être que l'état du déploiement reste `IN_PROGRESS` pendant une durée prolongée. La raison en est qu'Amazon ECS ne modifie pas le statut tant qu'il n'a pas supprimé le déploiement actif, et cela ne se produit qu'après la durée de l'intégration. En fonction de votre

configuration d'alarme, le déploiement peut sembler prendre plusieurs minutes de plus que lorsque vous n'utilisez pas d'alarmes (même si le nouvel ensemble de tâches principales est augmenté et l'ancien déploiement réduit). Si vous utilisez CloudFormation timeouts, pensez à augmenter les timeouts. Pour de plus amples informations, veuillez consulter Création de conditions d'attente dans un modèle dans le Guide de l'utilisateur AWS CloudFormation .

- Amazon ECS appelle `DescribeAlarms` pour sonder les alarmes. Les appels à `DescribeAlarms` comptez pour CloudWatch les quotas de service associés à votre compte. Si vous avez d'autres AWS services qui appellent `DescribeAlarms`, l'interrogation des alarmes par Amazon ECS peut avoir un impact. Par exemple, si un autre service effectue suffisamment d'appels à `DescribeAlarms` pour atteindre le quota, ce service est limité ; Amazon ECS est alors également limité et incapable d'interroger les alarmes. Si une alarme est générée pendant la période de limitation, Amazon ECS peut ne pas la détecter et la restauration peut ne pas avoir lieu. Cela n'a aucun autre impact sur le déploiement. Pour plus d'informations sur CloudWatch quotas de service, voir CloudWatch quotas de service dans le CloudWatch Guide de l'utilisateur.
- Si une alarme est activée à l'état `ALARM` au début d'un déploiement, Amazon ECS ne surveille pas les alarmes pendant la durée de ce déploiement (il ignore la configuration des alarmes). Ce comportement concerne le cas où vous souhaitez lancer un nouveau déploiement afin de corriger un échec initial.

Alarmes recommandées

Nous vous recommandons d'utiliser les métriques d'alarme suivantes :

- Si vous utilisez un Application Load Balancer, utilisez les métriques d'Application Load Balancer `HTTPCode_ELB_5XX_Count` et `HTTPCode_ELB_4XX_Count`. Ces métriques surveillent la présence de pics HTTP. Pour plus d'informations sur les mesures de l'équilibreur de charge des applications, consultez CloudWatch des mesures pour votre équilibreur de charge d'application dans le Guide de l'utilisateur pour les équilibreurs de charge d'application.
- Si vous avez déjà une application, utilisez les métriques `CPUUtilization` et `MemoryUtilization`. Elles vérifient l'utilisation de l'UC et de la mémoire en pourcentage utilisées par le cluster ou le service. Pour plus d'informations, consultez the section called “Considérations”.
- Si vous utilisez Amazon Simple Queue Service files d'attente dans vos tâches, utilisez `ApproximateNumberOfMessagesNotVisible` Métrique Amazon SQS. Cette métrique vérifie le nombre de messages dans la file d'attente qui sont retardés et qui ne peuvent pas être lus immédiatement. Pour plus d'informations sur les métriques Amazon SQS, consultez Disponible

CloudWatch statistiques pour Amazon SQS dans le Guide du développeur d'Amazon Simple Queue Service.

Durée de l’intégration

Lorsque vous utilisez l'option de restauration pour vos déploiements de services, Amazon ECS attend un délai supplémentaire après le déploiement de la révision du service cible avant d'envoyer un CloudWatch alarme. C’est ce que l’on appelle la durée de l’intégration. Cette durée commence lorsque :

- toutes les tâches d’une révision de service cible sont en cours d’exécution et en bon état ;
- les révisions du service source sont réduites à 0 %.

La durée de l’intégration par défaut est inférieure à 5 minutes. Le déploiement de service est marqué comme terminé après l’expiration de la durée de l’intégration.

Vous pouvez configurer la durée de l’intégration pour un déploiement continu. Quand vous utilisez CloudWatch alarmes pour détecter un échec. Si vous modifiez le temps de cuisson, puis décidez que vous souhaitez utiliser la valeur par défaut d'Amazon ECS, vous devez régler le temps de cuisson manuellement.

Hooks de cycle de vie pour les déploiements de service Amazon ECS

Lorsqu'un déploiement démarre, il passe par les différentes étapes de son cycle de vie. Chaque étape peut être dans un état tel que `IN_PROGRESS` ou `SUCCEEDED`. Vous pouvez configurer des hooks de cycle de vie à des étapes spécifiques pour exécuter une logique personnalisée ou introduire des points de décision avant le déploiement. Amazon ECS prend en charge deux types de hooks de cycle de vie :

Crochets Lambda

Amazon ECS invoque une fonction Lambda à des étapes spécifiques d'un déploiement. Votre fonction contient votre logique personnalisée et doit renvoyer un objet JSON contenant un `hookStatus` de `SUCCEEDED`, `FAILED`, ou `IN_PROGRESS` pour indiquer à Amazon ECS comment procéder. Vous pouvez utiliser ces crochets pour exécuter des tests de validation, appliquer des politiques de gouvernance ou implémenter des étapes d'approbation personnalisées avant le déploiement. Pour plus d’informations, consultez Hooks Lambda pour les déploiements de services Amazon ECS.

Crochets de pause

Amazon ECS interrompt le déploiement à une étape du cycle de vie configurée et attend que vous appeliez le `ContinueServiceDeployment` API pour continuer ou revenir en arrière. Pendant la pause, vous pouvez exécuter vos propres flux de travail en externe, tels que des approbations manuelles, des tests d'intégration avec des outils existants, des contrôles de préparation opérationnelle ou CI/CD étapes du pipeline. Pour plus d'informations, consultez Suspendre les hooks pour les déploiements de services Amazon ECS.

Vous pouvez configurer les hooks Lambda et les hooks pause au même stade du cycle de vie. Les deux hooks doivent être terminés avant que le déploiement ne passe à l'étape suivante.

Détails du cycle de vie des crochets

Lorsque les hooks sont actifs pendant un déploiement, vous pouvez consulter leur statut en appelant `DescribeServiceDeployments`. La réponse comprend un `lifecycleHookDetails` tableau contenant les champs suivants pour chaque hook actif :

Champ	Description
`hookId`	L'identifiant unique pour l'exécution de ce hook. Utilisez cette valeur lorsque vous appelez `ContinueServiceDeployment` pour les crochets de pause.
`targetType`	Type de crochet : `AWS_LAMBDA` ou `PAUSE`.
`targetArn`	L'ARN de la cible du hook. Pour les hooks Lambda, il s'agit de l'ARN de la fonction Lambda. Pour les crochets de pause, ce champ n'est pas défini.
`status`	État actuel du hook : `AWAITING_ACTION` , `IN_PROGRESS` , `SUCCEEDED` , `FAILED`, ou `TIMED_OUT` .
`expiresAt`	Date et heure d'expiration du hook. (Exemple : 2026-05-06T12:06:49-07:00)
`timeoutAction`	L'action qu'Amazon ECS entreprend lorsque le hook expire : `ROLLBACK` ou `CONTINUE`.

Catégories d'étapes du cycle de vie

Les étapes du cycle de vie entrent dans deux catégories principales :

1. Étapes d'invocation uniques - Amazon ECS invoque ces étapes une seule fois lors du déploiement d'un service :
   - `RECONCILE_SERVICE`
   - `PRE_SCALE_UP`
   - `POST_SCALE_UP`
   - `TEST_TRAFFIC_SHIFT`
   - `POST_TEST_TRAFFIC_SHIFT`
   - `POST_PRODUCTION_TRAFFIC_SHIFT`
2. Étapes d'invocation récurrentes - Amazon ECS peut invoquer ces étapes plusieurs fois lors du déploiement d'un service. Pour les déploiements linéaires et canaris, ces étapes sont invoquées à chaque étape de transfert de trafic :
   - `PRE_PRODUCTION_TRAFFIC_SHIFT`
   - `PRODUCTION_TRAFFIC_SHIFT`

> **Note**
>
> Les crochets de pause ne peuvent pas être configurés à `TEST_TRAFFIC_SHIFT` ou `PRODUCTION_TRAFFIC_SHIFT` car ces étapes sont également invoquées lors de la restauration. Une pause pendant un rollback nécessiterait une `ContinueServiceDeployment` appelez pour terminer la restauration.

Étapes prises en charge par type de crochet

Étape du cycle de vie	Crochets Lambda	Crochets de pause
`RECONCILE_SERVICE`	Oui	Oui
`PRE_SCALE_UP`	Oui	Oui
`POST_SCALE_UP`	Oui	Oui

Étape du cycle de vie	Crochets Lambda	Crochets de pause
TEST_TRAFFIC_SHIFT	Oui	Non
POST_TEST_TRAFFIC_SHIFT	Oui	Oui
PRE_PRODUCTION_TRAFFIC_SHIFT	Oui	Oui
PRODUCTION_TRAFFIC_SHIFT	Oui	Non
POST_PRODUCTION_TRAFFIC_SHIFT	Oui	Oui

Hooks Lambda pour les déploiements de services Amazon ECS

Les hooks Lambda sont des fonctions Lambda qu'Amazon ECS invoque à des étapes spécifiques d'un déploiement. Vous pouvez utiliser des hooks pour exécuter des tests de validation, appliquer des politiques de gouvernance ou implémenter une logique personnalisée avant le déploiement.

Lorsqu'Amazon ECS invoque votre hook, votre fonction doit renvoyer un objet JSON contenant un `hookStatus` champ. Vous pouvez éventuellement inclure `callBackDelay` pour contrôler le calendrier des nouvelles tentatives et `hookDetails` pour transmettre des données entre les invocations. Si votre fonction ne renvoie pas de valeur valide `hookStatus`, ou en cas d'échec, Amazon ECS annule le déploiement.

Valeurs HookStatus

Les éléments suivants sont valides `hookStatus` valeurs :

- `SUCCEEDED` - Le déploiement se poursuit jusqu'à la prochaine étape du cycle de vie.
- `FAILED` - Amazon ECS ramène le déploiement à la dernière révision de service réussie.
- `IN_PROGRESS` - Amazon ECS invoque à nouveau la fonction après un certain temps. Par défaut, le délai est de 30 secondes. Vous pouvez personnaliser cette valeur en renvoyant un `callBackDelay` aux côtés du `hookStatus`.

L'exemple suivant montre comment renvoyer un `hookStatus` avec un délai de rappel personnalisé. Dans cet exemple, Amazon ECS réessaie le hook après 60 secondes au lieu des 30 secondes par défaut :

```
{
    "hookStatus": "IN_PROGRESS",
    "callBackDelay": 60
}
```

État de passage avec HookDetails

Le `hookDetails` field est un dictionnaire que vous pouvez utiliser pour transmettre des données à votre fonction Lifecycle Hook. Il existe deux manières de remplir `hookDetails`:

- À la création ou à la mise à jour du service - Définissez `hookDetails` dans la configuration du hook du cycle de vie dans votre définition de service. Amazon ECS transmet ces données à votre fonction à chaque appel. Utilisez-le pour rendre vos hooks réutilisables sur plusieurs services en transmettant une configuration spécifique au service.
- Lors de l'exécution via les réponses IN_PROGRESS - Retour `hookDetails` aux côtés du `IN_PROGRESS` état du crochet. Amazon ECS renvoie ces données à votre fonction lors de la prochaine invocation. Utilisez-le pour maintenir l'état entre les appels sans stockage externe.

L'exemple suivant montre une configuration de hook de cycle de vie dans une définition de service qui transmet un nom de compartiment S3 à la fonction :

```
{
    "hookTargetArn": "arn:aws:lambda:us-west-2:123456789012:function:my-approval-hook",
    "roleArn": "arn:aws:iam::123456789012:role/ecs-lambda-invoke-role",
    "lifecycleStages": [
        "POST_TEST_TRAFFIC_SHIFT"
    ],
    "hookDetails": {
        "S3_BUCKET_NAME": "my-approval-bucket"
    }
}
```

Quand votre fonction revient `IN_PROGRESS`, vous pouvez également inclure `hookDetails` dans la réponse. Amazon ECS fusionne ces données et les renvoie lors de l'appel suivant. Les cas d'utilisation courants incluent le passage de compteurs de métriques ou d'ARN de ressources externes entre les invocations.

```
{
    "hookStatus": "IN_PROGRESS",
    "callBackDelay": 30,
    "hookDetails": {
        "approvalChecked": true,
        "S3_BUCKET_NAME": "my-approval-bucket"
    }
}
```

Lors de l'appel suivant, Amazon ECS inclut le `hookDetails` en cas de charge utile à côté `executionDetails`:

```
{
  "executionId": "e8d5a28f-eb01-4f3c-9454-a30ba6dc54bc",
  "lifecycleStage": "POST_TEST_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {}
  },
  "hookDetails": {
    "approvalChecked": true,
    "S3_BUCKET_NAME": "my-approval-bucket"
  }
}
```

Votre fonction peut lire `event["hookDetails"]` pour accéder à la configuration et à l'état des appels précédents.

> **Note**
>
> Données ajoutées à `hookDetails` au moment de l'exécution, elle ne persiste qu'entre les invocations du même hook au sein d'un même déploiement. Les données ne sont pas

transférées entre différents hooks au sein d'un même déploiement ou entre le même hook dans différents déploiements.

Configuration du délai d'attente

Vous pouvez configurer un délai d'expiration pour les hooks Lambda à l'aide de `timeoutConfiguration`. Si la fonction Lambda ne renvoie pas `SUCCEEDED` ou `FAILED` avant l'expiration du délai, Amazon ECS exécute l'action de temporisation configurée.

Champ	Description
`timeoutConfiguration.timeoutInMinutes`	Combien de temps attendre. Plage valide : 1 à 1 440 (24 heures). Par défaut : 1 440 minutes.
`timeoutConfiguration.action`	L'action à effectuer si le délai expire. Valeurs valides : `ROLLBACK`, `CONTINUE`. Valeur par défaut : `ROLLBACK`.

Étapes du cycle de vie prises

Vous pouvez configurer les hooks Lambda aux étapes du cycle de vie suivantes :

- `RECONCILE_SERVICE`
- `PRE_SCALE_UP`
- `POST_SCALE_UP`
- `TEST_TRAFFIC_SHIFT`
- `POST_TEST_TRAFFIC_SHIFT`
- `PRE_PRODUCTION_TRAFFIC_SHIFT`
- `PRODUCTION_TRAFFIC_SHIFT`
- `POST_PRODUCTION_TRAFFIC_SHIFT`

Pour les déploiements linéaires et Canary, les hooks Lambda sont configurés sur `PRODUCTION_TRAFFIC_SHIFT` ou `PRE_PRODUCTION_TRAFFIC_SHIFT` sont invoqués à chaque étape du changement de trafic.

Pour plus d'informations sur les catégories d'étapes du cycle de vie, voir Hooks de cycle de vie pour les déploiements de service Amazon ECS.

Données utiles de cycle de vie

Structure de données utiles communes

Lorsqu'Amazon ECS invoque la fonction Lambda de votre hook de cycle de vie, la charge utile de l'événement contient les champs de niveau supérieur suivants :

- `executionId` - L'identifiant unique pour l'exécution de ce hook.
- `lifecycleStage` - L'étape actuelle du cycle de vie (par exemple, `PRODUCTION_TRAFFIC_SHIFT`).
- `resourceArn` - L'ARN de la ressource associée au déploiement.
- `executionDetails` - Un objet contenant les informations spécifiques au déploiement décrites dans la liste suivante.

L'objet `executionDetails` comporte les champs suivants :

- `serviceArn` : l'Amazon Resource Name (ARN) du service.
- `targetServiceRevisionArn` : lÂRN de la révision de service cible est en cours de déploiement.
- `testTrafficWeights` : une carte des ARN de révision de services correspondant à leurs pourcentages de poids de trafic test.
- `productionTrafficWeights` : une carte des ARN de révision de services correspondant à leurs pourcentages de poids du trafic de production.

L'exemple suivant montre la structure complète des événements que reçoit votre fonction Lambda :

```
{
  "executionId": "f4fcae0f-9bec-41c6-ba87-0eaa0cef8af5",
  "lifecycleStage": "PRODUCTION_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
```

```
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/9313423515462893900": 100,
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/1920498462936580504": 0
    }
  }
}
```

Données utiles relatives aux étapes de cycle de vie

Les sections suivantes présentent des exemples de charges utiles pour chaque étape du cycle de vie. Dans ces exemples, la révision du service vert (9313423515462893900) est la nouvelle révision en cours de déploiement, et la révision du service bleu (1920498462936580504) est la révision de production existante.

PRE_SCALE_UP

Cette étape a lieu avant qu'Amazon ECS ne lance les tâches de révision du service vert. La révision du service vert n'a pas commencé et aucun trafic n'y est acheminé.

```
{
  "executionId": "e8d5a28f-eb01-4f3c-9454-a30ba6dc54bc",
  "lifecycleStage": "PRE_SCALE_UP",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {}
  }
}
```

POST_SCALE_UP

Cette étape se produit une fois qu'Amazon ECS a lancé les tâches de révision du service vert et que celles-ci sont en bon état. Les tâches vertes sont en cours d'exécution mais ne reçoivent pas encore de trafic.

```
{
  "executionId": "8b095b05-7bb0-4c56-a223-a3f61f4f9295",
  "lifecycleStage": "POST_SCALE_UP",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {}
  }
}
```

TEST_TRAFFIC_SHIFT

Cette étape se produit lorsqu'Amazon ECS déplace le trafic de test vers la révision du service vert. Le `testTrafficWeights` affiche la révision verte recevant 100 % du trafic de test tandis que la révision bleue reçoit 0 %. Le trafic de production continue d'affluer vers la version bleue.

```
{
  "executionId": "779085de-ab47-42bc-84ad-41f9914a8643",
  "lifecycleStage": "TEST_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/9313423515462893900": 100,
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/19204984629365805O4": 0
```

```
    },
    "productionTrafficWeights": {}
  }
}
```

POST_TEST_TRAFFIC_SHIFT

Cette étape a lieu une fois qu'Amazon ECS a terminé le test de transfert de trafic. La révision du service vert gère 100 % du trafic de test.

```
{
  "executionId": "3a0345ba-b029-404b-890d-7da2a4b266aa",
  "lifecycleStage": "POST_TEST_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {}
  }
}
```

CHANGEMENT_DE_CIRCULATION_PRÉPRODUCTION

Cette étape a lieu avant qu'Amazon ECS ne transfère le trafic de production vers la révision du service vert. Pour les déploiements linéaires et Canary, cette étape est invoquée avant chaque étape de transfert de trafic.

```
{
  "executionId": "a2b3c4d5-e6f7-8901-abcd-ef1234567890",
  "lifecycleStage": "PRE_PRODUCTION_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
```

```
    "testTrafficWeights": {},
    "productionTrafficWeights": {
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/9313423515462893900": 0,
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/1920498462936580504": 100
    }
  }
}
```

PRODUCTION_TRAFFIC_SHIFT

Cette étape se produit lorsqu'Amazon ECS déplace le trafic de production vers la révision du service vert. Le `productionTrafficWeights` affiche la révision verte recevant 100 % du trafic de production tandis que la révision bleue reçoit 0 %.

```
{
  "executionId": "f4fcae0f-9bec-41c6-ba87-0eaa0cef8af5",
  "lifecycleStage": "PRODUCTION_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/9313423515462893900": 100,
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/1920498462936580504": 0
    }
  }
}
```

POST_PRODUCTION_TRAFFIC_SHIFT

Cette étape a lieu une fois qu'Amazon ECS a terminé le transfert du trafic de production. La révision du service vert gère désormais tout le trafic de production.

```
{
  "executionId": "5f40ed04-7e54-437d-b95d-98bc872fec49",
  "lifecycleStage": "POST_PRODUCTION_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
    "productionTrafficWeights": {}
  }
}
```

Hooks liés au cycle de vie lors de la restauration

Il n'y a pas de dédié `ROLLBACK` étape du cycle de vie. Lorsqu'un rollback se produit, Amazon ECS invoque à nouveau les hooks enregistrés sur le `PRODUCTION_TRAFFIC_SHIFT` et `TEST_TRAFFIC_SHIFT` étapes (les étapes d'invocation récurrentes). Lors d'une restauration, `productionTrafficWeights` dans la charge utile, le trafic revient à la version bleue.

Le `targetServiceRevisionArn` reste l'ARN de révision vert car il constitue toujours la cible du déploiement d'origine, même si le trafic s'en éloigne.

L'exemple suivant montre un `PRODUCTION_TRAFFIC_SHIFT` charge utile lors d'un rollback. Notez que la version bleue (`1920498462936580504`) reçoit désormais 100 % du trafic de production, tandis que la version verte (`9313423515462893900`) reçoit 0 % :

```
{
  "executionId": "70073435-cb99-457f-b900-6ee1dcad05ec",
  "lifecycleStage": "PRODUCTION_TRAFFIC_SHIFT",
  "resourceArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/EZe5RNVLH6PPzHXINuP28",
  "executionDetails": {
    "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-service",
    "targetServiceRevisionArn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
my-cluster/my-service/9313423515462893900",
    "testTrafficWeights": {},
```

```
    "productionTrafficWeights": {
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/9313423515462893900": 0,
      "arn:aws:ecs:us-west-2:123456789012:service-revision/my-cluster/my-
service/19204984629365805O4": 100
    }
  }
}
```

Pour déterminer si votre hook est invoqué lors d'un rollback, vérifiez le `productionTrafficWeights`. Si le `targetServiceRevisionArn` (révision verte) a un poids de 0 % et l'autre révision est de 100 %, le déploiement est en cours d'annulation.

Suspendre les hooks pour les déploiements de services Amazon ECS

Les crochets de pause interrompent le déploiement d'un service Amazon ECS à un stade de cycle de vie spécifique et attendent que vous appeliez `ContinueServiceDeployment` avant le début du déploiement. Utilisez les crochets de pause lorsque vous avez besoin d'une approbation manuelle ou d'une validation externe avant la poursuite d'un déploiement.

Comment fonctionnent les crochets de pause

Lorsqu'un déploiement atteint une étape du cycle de vie pour laquelle un crochet de pause est configuré, les événements suivants se produisent :

1. Amazon ECS génère une solution unique `hookId` pour le crochet de pause.
2. Amazon ECS émet un EventBridge événement avec le type de détail `ECS Hook State Change` et le nom de l'événement `HOOK_AWAITING_ACTION`.
3. Le déploiement reste suspendu jusqu'à ce que vous appeliez `ContinueServiceDeployment` avec le `hookId` et une action de `CONTINUE` ou `ROLLBACK`, ou jusqu'à ce que le délai d'expiration configuré soit atteint.

Vous pouvez récupérer le `hookId` en appelant `DescribeServiceDeployments`. La réponse comprend un `lifecycleHookDetails` tableau avec l'état du hook :

```
{
    "serviceDeployments": [
```

```
        {
            "lifecycleHookDetails": [
                {
                    "hookId": "ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw",
                    "targetType": "PAUSE",
                    "status": "AWAITING_ACTION",
                    "expiresAt": "2024-01-15T12:00:00Z",
                    "timeoutAction": "ROLLBACK"
                }
            ]
        }
    ]
}
```

Configuration des crochets de pause

L'exemple suivant montre une configuration de crochet de pause dans une définition de service qui interrompt le déploiement une fois le changement de trafic de test terminé :

```
{
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "lifecycleHooks": [
            {
                "targetType": "PAUSE",
                "lifecycleStages": [
                    "POST_TEST_TRAFFIC_SHIFT"
                ],
                "timeoutConfiguration": {
                    "timeoutInMinutes": 10,
                    "action": "ROLLBACK"
                }
            }
        ]
    }
}
```

Champs de configuration

Champ	Description	Valeurs valides	Obligatoire
`targetType`	Type de hook. Doit être `PAUSE` pour les crochets de pause.	`PAUSE`	Oui
`lifecycleStages`	Les étapes du cycle de vie au cours desquelles le déploiement s'arrête.	`RECONCILE_SERVICE`, `PRE_SCALE_UP`, `POST_SCALE_UP`, `POST_TEST_TRAFFIC_SHIFT`, `PRE_PRODUCTION_TRAFFIC_SHIFT`, `POST_PRODUCTION_TRAFFIC_SHIFT`	Oui
`timeoutConfiguration.timeoutInMinutes`	Combien de temps faut-il attendre avant d'effectuer l'action de temporisation. Par défaut : 1 440 minutes (24 heures).	1 à 20 160 (14 jours)	Non
`timeoutConfiguration.action`	L'action à effectuer si le délai expire.	`ROLLBACK` (par défaut), `CONTINUE`	Non

Contraintes

- Vous pouvez configurer un maximum de 10 crochets de pause et 10 crochets Lambda par service.

- Les crochets Pause ne sont pas utilisés `hookTargetArn` ou `roleArn`. Ces champs concernent uniquement les hooks Lambda.
- Les crochets de pause ne peuvent pas être configurés sur `TEST_TRAFFIC_SHIFT` ou `PRODUCTION_TRAFFIC_SHIFT` étapes.

Poursuivre un déploiement suspendu

Lorsqu'un déploiement est suspendu (l'état du crochet de pause est `AWAITING_ACTION`), appelez `ContinueServiceDeployment` avec le `hookId` et l'action que vous souhaitez entreprendre.

Pour poursuivre le déploiement, procédez comme suit :

```
aws ecs continue-service-deployment \
    --hook-id ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw \
    --action CONTINUE
```

Pour annuler le déploiement :

```
aws ecs continue-service-deployment \
    --hook-id ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw \
    --action ROLLBACK
```

Pour plus d’informations, consultez Déploiements de services continus.

Plusieurs crochets au même stade

Lorsque plusieurs hooks sont configurés au même stade du cycle de vie, le déploiement reste suspendu jusqu'à ce que tous les hooks à ce stade soient poursuivis. Si un hook déclenche un rollback, l'ensemble du déploiement est annulé quel que soit le statut des autres hooks à ce stade.

Ce comportement s'applique lorsque les hooks Lambda et les hooks pause sont configurés au même stade du cycle de vie. Le déploiement ne se poursuit que lorsque tous les hooks à ce stade se sont terminés avec succès ou ont été poursuivis.

Tous les hooks configurés au même stade du cycle de vie s'exécutent en parallèle.

Comportement en cas d'expiration

Si vous n'appelez pas `ContinueServiceDeployment` avant l'expiration du délai, Amazon ECS exécute l'action de temporisation configurée :

- `ROLLBACK` (par défaut) - Amazon ECS rétablit le déploiement à la révision de service précédente.
- `CONTINUE` - Amazon ECS poursuit le déploiement jusqu'à la prochaine étape du cycle de vie.

Vérifiez le `expiresAt` sur le terrain `lifecycleHookDetails` tableau des `DescribeServiceDeployments` réponse pour voir quand le délai expire.

Suspendez les hooks avec des déploiements linéaires et canariens

Pour les déploiements linéaires et Canary, suspendez les hooks configurés sur `PRE_PRODUCTION_TRAFFIC_SHIFT` sont invoqués à chaque étape du changement de trafic. Chaque invocation génère une valeur unique `hookId`, et vous devez appeler `ContinueServiceDeployment` pour chacun d'entre eux.

Par exemple, un déploiement linéaire avec 5 étapes de changement de trafic et un crochet de pause à `PRE_PRODUCTION_TRAFFIC_SHIFT` fait des pauses 5 fois - une fois avant chaque étape. Chaque pause nécessite un appel distinct à `ContinueServiceDeployment` avec le correspondant `hookId`.

EventBridge événements

Amazon ECS émet EventBridge événements avec le type de détail `ECS Hook State Change` à mesure que les crochets de pause progressent dans leur cycle de vie. Le tableau suivant décrit les noms des événements :

Nom de l'événement	Description
`HOOK_AWAITING_ACTION`	Le hook attend que vous appeliez `ContinueServiceDeployment` .
`HOOK_SUCCEEDED`	Le hook s'est terminé avec succès (vous avez appelé `ContinueServiceDeployment` avec `CONTINUE`).
`HOOK_FAILED`	Le hook a échoué (vous avez appelé `ContinueServiceDeployment` avec `ROLLBACK`).

Nom de l'événement	Description
HOOK_TIMED_OUT	Le crochet a expiré avant ContinueServiceDeployment a été appelé.

L'exemple suivant montre un EventBridge événement émis lorsqu'un hook de pause attend une action :

```
{
    "version": "0",
    "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
    "detail-type": "ECS Hook State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2024-01-15T10:00:00Z",
    "region": "us-west-2",
    "detail": {
        "eventType": "INFO",
        "eventName": "HOOK_AWAITING_ACTION",
        "hookId": "ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw",
        "hookType": "PAUSE",
        "expiresAt": "2024-01-15T12:00:00Z",
        "lifecycleStage": "POST_TEST_TRAFFIC_SHIFT",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-
service",
        "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/
my-cluster/my-service/EZe5RNVLH6PPzHXINuP28",
        "updatedAt": "2024-01-15T10:00:00Z"
    }
}
```

Déploiements de services continus

Lorsqu'un hook de pause du cycle de vie est configuré pour un déploiement de service Amazon ECS, le déploiement s'arrête au stade du cycle de vie configuré et attend une action explicite. Utilisez le `ContinueServiceDeployment` API permettant de poursuivre le déploiement à l'étape suivante du cycle de vie ou de revenir à la révision de service précédente.

Les types d'actions suivants sont disponibles :

- `CONTINUE` - Cette option permet de poursuivre le déploiement jusqu'à l'étape suivante du cycle de vie.
- `ROLLBACK` - Cette option ramène le déploiement du service à la révision précédente du service.

Vous pouvez poursuivre un déploiement qui comporte un crochet de pause `AWAITING_ACTION` statut. Le déploiement doit se faire dans le `IN_PROGRESS` état. Pour de plus amples informations sur les états de déploiement de service, consultez la section Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Conditions préalables

- Le crochet de pause doit être activé `AWAITING_ACTION` statut.
- Vous avez besoin du `hookId` à partir du `DescribeServiceDeployments` réponse.

Procédure

ECS Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Dans le Des services section, choisissez le service.
4. Choisissez l'onglet Déploiements.
5. Vérifiez que l'état du crochet de pause s'affiche En attente d'action.
6. Choisissez Passez à l'action, puis choisissez Continuer ou Revenir en arrière.

CLI

1. Obtenez le `hookId` en appelant `describe-service-deployments`:

```
aws ecs describe-service-deployments \
    --service-deployment-arn arn:aws:ecs:us-west-2:123456789012:service-
deployment/my-cluster/my-service/EZe5RNVLH6PPzHXINuP28
```

2. Appel `continue-service-deployment` avec le `hookId` et l'action :

   Pour poursuivre le déploiement, procédez comme suit :

   ```
   aws ecs continue-service-deployment \
       --hook-id ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw \
       --action CONTINUE
   ```

   Pour annuler le déploiement :

   ```
   aws ecs continue-service-deployment \
       --hook-id ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw \
       --action ROLLBACK
   ```

   La réponse inclut l'ARN de déploiement du service :

   ```
   {
       "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
   deployment/my-cluster/my-service/EZe5RNVLH6PPzHXINuP28"
   }
   ```

Étapes suivantes

- Surveillez la progression du déploiement à l'aide de `DescribeServiceDeployments` ou la console Amazon ECS.
- Si des crochets de pause supplémentaires sont configurés à des étapes ultérieures du cycle de vie, le déploiement s'arrête à nouveau à ces étapes.

Arrêt de déploiements de service Amazon ECS

Vous pouvez arrêter manuellement un déploiement lorsqu'aucun déploiement défaillant n'a été détecté par le disjoncteur ou CloudWatch alarmes. Les types d’arrêt suivants sont disponibles :

- Restauration : cette option restaure le déploiement de service à la version précédente du service.

  Vous pouvez utiliser cette option même si vous n'avez pas configuré le déploiement de service pour l'option de restauration.

Vous pouvez arrêter un déploiement qui se trouve dans l'un des états suivants. Pour de plus amples informations sur les états de déploiement de service, consultez la section Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

- EN ATTENTE : le déploiement de service passe à l'état RESTAURATION_DEMAMDÉE, puis l'opération de restauration commence.
- EN_COURS : le déploiement de service passe à l'état RESTAURATION_DEMAMDÉE, puis l'opération de restauration commence.
- ARRÊT_DEMANDÉ : le déploiement de service continue de s'arrêter.
- RESTAURATION_DEMAMDÉE : le déploiement de service poursuit l'opération de restauration.
- RESTAURATION_EN_COURS : le déploiement de service poursuit l'opération de restauration.

Procédure

Avant de commencer, configurez les autorisations requises pour consulter les déploiements de service. Pour plus d'informations, consultez Autorisations requises pour consulter les déploiements de service Amazon ECS.

Amazon ECS Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s'affiche.
4. Sur la page des détails du service, choisissez Déploiements.

   La page des déploiements s'affiche.
5. Sous Déploiement en cours, choisissez Restaurer. Ensuite, dans la fenêtre de confirmation, choisissez Restaurer.

AWS CLI

1. Exécutez `list-service-deployments` pour récupérer l'ARN de déploiement de service.

   Remplacez le *user-input* avec tes valeurs.

   ```
   aws ecs list-service-deployments --cluster cluster-name --service service-name
   ```

   Notez le `serviceDeploymentArn` du déploiement que vous souhaitez arrêter.

   ```
   {
       "serviceDeployments": [
           {
               "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
   deployment/cluster-name/service-name/NCWGC2ZR-taawPAYrIaU5",
               "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/cluster-
   name/service-name",
               "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/cluster-
   name",
               "targetServiceRevisionArn": "arn:aws:ecs:us-
   west-2:123456789012:service-revision/cluster-name/service-
   name/4980306466373577095",
               "status": "SUCCESSFUL"
           }
       ]
   }
   ```

2. Exécutez `stop-service-deployments`. Utilisez celui `serviceDeploymentArn` qui a été renvoyé par `list-service-deployments`.

   Remplacez le *user-input* avec tes valeurs.

   ```
   aws ecs stop-service-deployment --service-deployment-arn
    arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
   name/NCWGC2ZR-taawPAYrIaU5 --stop-type ROLLBACK
   ```

Étapes suivantes

Décidez quelles modifications doivent être apportées au service, puis mettez-le à jour. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Déploiement de services Amazon ECS en remplaçant des tâches

Lorsque vous créez un service qui utilise le type de déploiement Mise à jour propagée (ECS), le planificateur de service Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

Amazon ECS utilise les paramètres suivants pour déterminer le nombre de tâches :

- Le `minimumHealthyPercent` représente la limite inférieure du nombre de tâches qui doivent être en cours d'exécution et en bon état pour un service lors d'un déploiement continu ou lorsqu'une instance de conteneur est épuisée, en pourcentage du nombre de tâches souhaité pour le service. Cette valeur est arrondie à la valeur supérieure. Par exemple, si le pourcentage minimum d'instances saines est `50` et que le nombre de tâches souhaité est de quatre, le planificateur peut arrêter deux tâches existantes avant de démarrer deux nouvelles tâches. De même, si le pourcentage de santé minimum est de 75 % et que le nombre de tâches souhaité est de deux, le planificateur ne peut pas arrêter de tâche car la valeur résultante est également de deux.
- Le `maximumPercent` représente la limite supérieure du nombre de tâches qui doivent être exécutées pour un service lors d'un déploiement continu ou lorsqu'une instance de conteneur est épuisée, en pourcentage du nombre de tâches souhaité pour un service. Cette valeur est arrondie à la valeur inférieure. Par exemple, si le pourcentage maximum est `200` et si le nombre de tâches souhaité est de quatre, le planificateur peut démarrer quatre nouvelles tâches avant d'arrêter quatre tâches existantes. De même, si le pourcentage maximal est `125` et que le nombre de tâches souhaité est de trois, le planificateur ne peut pas démarrer de tâche, car la valeur résultante est également de trois.

Au cours d'un déploiement continu, lorsque les tâches deviennent inefficaces, Amazon ECS les remplace pour maintenir les `minimumHealthyPercent` et protégez la disponibilité. Les tâches défectueuses sont remplacées par la même révision de service à laquelle elles appartiennent. Cela garantit que le remplacement incorrect des tâches dans la révision source est indépendant des échecs de tâches dans la révision cible. Quand le `maximumPercent` si ce paramètre le permet, le planificateur lance les tâches de remplacement avant d'arrêter celles qui ne sont pas saines. Si le `maximumPercent` Le paramètre empêche le planificateur de démarrer une tâche de remplacement en premier, le planificateur arrête une tâche défectueuse à la fois pour libérer de la capacité avant de lancer une tâche de remplacement.

> **Important**
>
> Lorsque vous définissez un pourcentage minimum ou maximal d'instances saines, vous devez vous assurer que le planificateur peut arrêter ou démarrer au moins une tâche lorsqu'un déploiement est déclenché. Si votre service a un déploiement bloqué en raison d'une configuration de déploiement non valide, un message d'événement de service est envoyé. Pour plus d'informations, consultez service (*service-name*) n'a pas pu arrêter ou démarrer des tâches pendant un déploiement en raison de la configuration de déploiement du service. Mettre à jour le minimumHealthyPercent ou la valeur MaximumPercent et réessayez..

Les déploiements progressifs s'appuient sur deux méthodes permettant d'identifier rapidement l'échec d'un déploiement de service :

- the section called " Comment le disjoncteur de déploiement détecte les pannes"
- the section called "Comment CloudWatch les alarmes détectent les échecs de déploiement"

Les méthodes peuvent être utilisées séparément ou conjointement. Lorsque les deux méthodes sont utilisées, le déploiement est défini en échec dès que les critères d'échec de l'une ou l'autre méthode sont remplis.

Suivez les instructions suivantes pour déterminer quelle méthode utiliser :

- Disjoncteur : utilisez cette méthode lorsque vous souhaitez arrêter un déploiement car les tâches ne peuvent pas démarrer.
- CloudWatch alarmes : utilisez cette méthode lorsque vous souhaitez arrêter un déploiement en fonction des métriques de l'application.

Les deux méthodes permettent de restaurer la version de service précédente.

Vous pouvez configurer des critères de réussite précoces pour qu'Amazon ECS réalise un déploiement continu lorsqu'un pourcentage des tâches que vous définissez est valide dans la révision du service cible. Pour plus d'informations, consultez Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces.

Résolution des images de conteneurs

Amazon ECS résout les noms des images de conteneurs et toutes les balises d'image spécifiées dans la définition de la tâche en condensés d'images de conteneurs. Si vous créez un service qui exécute et gère une seule tâche, cette tâche est utilisée pour établir des condensés d'images pour les conteneurs de la tâche. Si vous créez un service qui exécute et gère plusieurs tâches, la première tâche démarrée par le planificateur de service lors du déploiement est utilisée pour établir les condensés d'images pour les conteneurs contenus dans les tâches.

Si au moins trois tentatives d'établissement des condensés d'images du conteneur échouent, le déploiement se poursuit sans résolution de condensé d'image. Si le disjoncteur de déploiement est activé, le déploiement échoue également et est annulé.

Une fois les condensés des images du conteneur établis, Amazon ECS les utilise pour démarrer toutes les autres tâches souhaitées et pour toute future mise à jour de service. Ainsi, toutes les tâches d'un service exécutent toujours des images de conteneur identiques, ce qui garantit la cohérence des versions de votre logiciel.

Vous pouvez configurer ce comportement pour chaque conteneur de votre tâche en utilisant le paramètre `versionConsistency` figurant dans la définition du conteneur. Pour plus d'informations, consultez `versionConsistency`.

> **Note**
>
> - Les versions de l'agent Amazon ECS inférieures à `1.31.0` ne prennent pas en charge la résolution du condensé d'image. Les versions de l'agent `1.31.0` à `1.69.0` ne prenent en charge la résolution de condensé d'images que pour les images transférées vers les référentiels Amazon ECR. Les versions de l'agent `1.70.0` ou supérieures prennent en charge la résolution de condensé d'image pour toutes les images.
> - La version de plateforme Linux Fargate minimale pour la résolution du résumé d'image est `1.3.0`. La version de plateforme Windows Fargate minimale pour la résolution du condensé d'image est `1.0.0`.
> - Amazon ECS ne capture pas les résumés des conteneurs sidecar gérés par Amazon ECS, tels que Amazon GuardDuty agent de sécurité ou proxy Service Connect.
> - Pour réduire la latence potentielle associée à la résolution des images de conteneur dans les services comportant plusieurs tâches, exécutez la version `1.83.0` ou supérieure de l'agent Amazon ECS sur les instances de conteneur EC2. Pour éviter toute latence potentielle, spécifiez des condensés d'images de conteneur dans votre définition de tâche.

- Si vous créez un service avec un nombre de tâches souhaité égal à zéro, Amazon ECS ne peut pas établir de résumés de conteneurs tant que vous n'avez pas déclenché un autre déploiement du service avec un nombre de tâches souhaité supérieur à zéro.
- Pour établir des condensés d'images mis à jour, vous pouvez forcer un nouveau déploiement. Les condensés mis à jour seront utilisés pour démarrer de nouvelles tâches et n'affecteront pas les tâches déjà en cours. Pour plus d'informations sur la manière de forcer de nouveaux déploiements, consultez forceNewDeployment dans le Référence de l'API Amazon ECS.
- Lorsque vous utilisez des fournisseurs de capacité EC2, si la capacité est insuffisante pour démarrer une tâche lors du déploiement initial, la cohérence des versions logicielles peut échouer. Pour garantir la cohérence des versions même lorsque la capacité est limitée, définisse explicitement `versionConsistency: "enabled"` dans la configuration de votre conteneur de définition de tâches plutôt que de vous fier au comportement par défaut. Cela oblige Amazon ECS à attendre que la capacité soit disponible avant de procéder au déploiement.

Pratiques exemplaires relatives aux paramètres d'un service Amazon ECS

Pour éviter toute durée d'indisponibilité de l'application, le processus de déploiement est le suivant :

1. Démarrez les nouveaux conteneurs d'applications tout en maintenant les conteneurs existants en activité.
2. Vérifiez que les nouveaux conteneurs sont sains.
3. Arrêtez les anciens conteneurs.

En fonction de la configuration de votre déploiement et de la quantité d'espace libre non réservé dans votre cluster, plusieurs cycles peuvent être nécessaires pour remplacer toutes les anciennes tâches par de nouvelles tâches.

Il existe deux options de configuration de service que vous pouvez utiliser pour modifier le nombre :

- `minimumHealthyPercent` : 100 % (par défaut)

  La limite inférieure du nombre de tâches pour lesquelles votre service doit rester dans l'état `RUNNING` lors d'un déploiement. Cette valeur est exprimée en tant que pourcentage de

`desiredCount` arrondi à la valeur supérieure la plus proche. Ce paramètre vous permet de procéder au déploiement sans avoir recours à une capacité de cluster supplémentaire.

- `maximumPercent` : 200 % (par défaut)

  La limite supérieure du nombre de tâches pour votre service autorisées dans l'état `RUNNING` ou `PENDING` pendant un déploiement. Cette valeur est exprimée en tant que pourcentage de `desiredCount` arrondi à la valeur inférieure la plus proche.

Exemple : options de configuration par défaut

Considérons le service suivant qui comporte six tâches, déployées dans un cluster pouvant accueillir huit tâches au total. Les options de configuration de service par défaut ne permettent pas au déploiement d'aller en dessous de 100 % des six tâches souhaitées.

Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les six tâches.
2. Le planificateur lance deux nouvelles tâches, car les paramètres par défaut exigent que six tâches soient en cours d'exécution.

   Il existe désormais six tâches existantes et deux nouvelles tâches.
3. Le planificateur arrête deux des tâches existantes.

   Il existe désormais quatre tâches existantes et deux nouvelles tâches.
4. Le planificateur lance deux nouvelles tâches supplémentaires.

   Il existe désormais quatre tâches existantes et quatre nouvelles tâches.
5. Le planificateur arrête deux des tâches existantes.

   Il existe désormais deux tâches existantes et quatre nouvelles.
6. Le planificateur lance deux nouvelles tâches supplémentaires.

   Il y a maintenant deux tâches existantes et six nouvelles tâches
7. Le planificateur arrête les deux dernières tâches existantes.

   Il y a désormais six nouvelles tâches.

Dans l'exemple précédent, si vous utilisez les valeurs par défaut pour les options, il y a une attente de 2,5 minutes pour chaque nouvelle tâche qui démarre. En outre, l'équilibreur de charge devra peut-être attendre 5 minutes pour que l'ancienne tâche s'arrête.

Exemple : modifier **`minimumHealthyPercent`**

Vous pouvez accélérer le déploiement en définissant la valeur `minimumHealthyPercent` sur 50 %.

Considérons le service suivant qui comporte six tâches, déployées dans un cluster pouvant accueillir huit tâches au total. Le processus de déploiement est le suivant :

1. L'objectif est de remplacer six tâches.
2. Le planificateur arrête trois des tâches existantes.

   Il subsiste trois tâches en cours d'exécution qui correspondent à la valeur de `minimumHealthyPercent`.
3. Le planificateur lance cinq nouvelles tâches.

   Il existe trois tâches existantes et cinq nouvelles tâches.
4. Le planificateur arrête les trois tâches existantes restantes.

   Il y a cinq nouvelles tâches
5. Le planificateur lance la dernière nouvelle tâche.

   Il y a six nouvelles tâches.

Exemple : modifier l'espace libre du cluster

Vous pouvez également ajouter de l'espace libre afin de pouvoir exécuter des tâches supplémentaires.

Considérons le service suivant qui comporte six tâches, déployées dans un cluster pouvant accueillir dix tâches au total. Le processus de déploiement est le suivant :

1. L'objectif est de remplacer les tâches existantes.
2. Le planificateur arrête trois des tâches existantes.

   Il y a trois tâches existantes.
3. Le planificateur lance six nouvelles tâches.

Il y a maintenant trois tâches existantes et six nouvelles tâches

4. Le planificateur arrête les trois tâches existantes.

   Il y a six nouvelles tâches.

Recommandations

Utilisez les valeurs suivantes pour les options de configuration du service lorsque vos tâches sont inactives pendant un certain temps et que leur taux d'utilisation n'est pas élevé.

- `minimumHealthyPercent` : 50 %
- `maximumPercent` : 200 %

Création d'un déploiement de mise à jour propagée Amazon ECS

Créez un service pour exécuter et maintenir simultanément un nombre spécifié d'instances d'une définition de tâche dans un cluster. Si l'une de vos tâches échoue ou s'arrête, le planificateur de service d'Amazon ECS service lance une autre instance de votre définition de tâche pour la remplacer. Cela permet de maintenir le nombre de tâches souhaité dans le service.

Définissez les paramètres de configuration suivants avant de créer un service :

- Deux options de calcul permettent de distribuer vos tâches.
  - Une capacity provider strategy (stratégie de fournisseurs de capacités) permet à Amazon ECS de distribuer vos tâches avec un ou plusieurs fournisseurs de capacité.

    Si vous souhaitez exécuter vos charges de travail sur des instances gérées Amazon ECS, vous devez utiliser l'option de stratégie de fournisseur de capacité.
  - Un type de lancement permet à Amazon ECS de lancer vos tâches directement sur Fargate ou sur les instances Amazon EC2 enregistrées dans vos clusters.

    Si vous souhaitez exécuter vos charges de travail sur des instances gérées Amazon ECS, vous devez utiliser l'option de stratégie de fournisseur de capacité.
- Les définitions de tâches qui utilisent le mode réseau `awsvpc` ou les services configurés pour utiliser un équilibreur de charge doivent avoir une configuration réseau. Par défaut, la console sélectionne l'Amazon VPC par défaut ainsi que tous ses sous-réseaux et son groupe de sécurité par défaut.

- La stratégie de placement des tâches par défaut répartit les tâches de manière uniforme entre les zones de disponibilité.

  Nous vous recommandons d'utiliser le rééquilibrage des zones de disponibilité pour garantir la haute disponibilité de votre service. Pour plus d'informations, consultez Équilibrage d'un service Amazon ECS entre les zones de disponibilité.
- Lorsque vous utilisez Launch Type (Type de lancement) pour le déploiement de votre service, le service démarre par défaut dans les sous-réseaux de votre VPC de cluster.
- Pour la stratégie des fournisseurs de capacités, la console sélectionne une option de calcul par défaut. Voici l'ordre utilisé par la console pour sélectionner une valeur par défaut :
  - Si votre cluster a une stratégie de fournisseur de capacité définie par défaut, elle est sélectionnée.
  - Si votre cluster n'a pas de stratégie de fournisseur de capacité par défaut définie, mais que les fournisseurs de capacité Fargate ont été ajoutés au cluster, une stratégie de fournisseur de capacité personnalisée qui utilise le fournisseur de capacité `FARGATE` est sélectionnée.
  - Si votre cluster ne dispose pas d'une stratégie de fournisseur de capacité par défaut définie, mais qu'un ou plusieurs fournisseurs de capacité de groupe Auto Scaling ont été ajoutés au cluster, l'option Utiliser une stratégie personnalisée (avancée) est sélectionnée et vous devez définir manuellement la stratégie.
  - Si votre cluster n'a pas de stratégie de fournisseur de capacité par défaut définie et qu'aucun fournisseur de capacité n'est ajouté au cluster, le type de lancement Fargate est sélectionné.
- Les options par défaut de détection des échecs de déploiement consistent à utiliser l'option Disjoncteur de déploiement Amazon ECS avec l'option Restauration en cas d'échec.

  Pour plus d'informations, consultez Détection des pannes par le disjoncteur de déploiement Amazon ECS.
- Décidez si vous souhaitez qu'Amazon ECS augmente ou diminue automatiquement le nombre souhaité de tâches dans votre service. Pour plus d'informations, consultez Mise à l'échelle automatique de votre service Amazon ECS.
- Si vous avez besoin d'une application pour vous connecter à d'autres applications qui s'exécutent dans Amazon ECS, déterminez l'option adaptée à votre architecture. Pour plus d'informations, consultez Interconnexion des services Amazon ECS.
- Lorsque vous créez un service qui utilise le disjoncteur Amazon ECS, Amazon ECS crée un déploiement et une révision de service. Ces ressources vous permettent de consulter des

informations détaillées sur l'historique des services. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Pour plus d'informations sur la création d'un service à l'aide du AWS CLI, voir create-service dans le AWS Command Line Interface Référence.

Pour plus d'informations sur la création d'un service à l'aide de AWS CloudFormation, voir AWS::ECS::Service dans le AWS CloudFormation Guide de l'utilisateur.

Création d'un service avec les options par défaut

Vous pouvez utiliser la console pour créer et déployer rapidement un service. Le service a la configuration suivante :

- Déploie dans le VPC et les sous-réseaux associés à votre cluster
- Déploie une tâche
- Utilise le déploiement continu
- Utilise la stratégie de fournisseur de capacité avec votre fournisseur de capacité par défaut
- Utilise le disjoncteur de déploiement pour détecter les défaillances et définit l'option permettant d'annuler automatiquement le déploiement en cas de problème

Pour déployer un service à l'aide des paramètres par défaut, procédez comme suit.

Pour créer un service (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la page de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster dans lequel créer le service.
4. Sous l'onglet Services choisissez Create (Créer).

   La page Créer un service s'affiche.
5. Sous Détails du service, procédez comme suit :

   a. Pour Définition de tâche, saisissez la famille et la révision de définition de tâche à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
6. Pour utiliser ECS Exec afin de déboguer le service, sous Configuration de la résolution des problèmes, sélectionnez Activer ECS Exec.

7. Sous Configuration du déploiement, procédez comme suit :

   - Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

8. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

   Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Task definitions (Définitions de tâches).

   Pour qu'Amazon ECS étiquette automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Turn on Amazon ECS managed tags (Activer les balises gérées par Amazon ECS), puis sélectionnez Service.

   Ajoutez ou supprimez une balise.

   - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
     - Pour Clé, saisissez le nom de la clé.
     - Pour Valeur, saisissez la valeur de clé.
   - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

Créer un service à l'aide de paramètres définis

Pour créer un service à l’aide de paramètres définis, procédez comme suit.

Pour créer un service (console Amazon ECS)

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes	
Clusters	a. Sur la page Clusters, sélectionnez le cluster	

Pour démarrer un service à partir de	Étapes
	dans lequel créer le service. b. Sous l'onglet Services choisissez Create (Créer).
Définition de tâche	a. Sur la page Définitions de tâches, sélectionnez le bouton d'option situé à côté de la définition de tâche. b. Dans le menu Déployer, choisissez Créer un service.

La page Créer un service s'affiche.

3. Sous Détails du service, procédez comme suit :

   a. Pour Définition de tâche, saisissez la définition de tâche à utiliser. Ensuite, pour Révision, choisissez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.

4. Pour Cluster existant, choisissez le cluster.

   Choisissez Créer un cluster pour exécuter la tâche sur un nouveau cluster

5. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Sous Configuration de calcul, choisissez votre option.

Options de calcul	Étapes
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie :

Options de calcul	Étapes
	• Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids. **Note** Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit

Options de calcul	Étapes
	être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque vous utilisez Fargate, pour Version de plateforme, spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

6. Pour utiliser ECS Exec afin de déboguer le service, sous Configuration de la résolution des problèmes, sélectionnez Activer ECS Exec.
7. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Service type (Type de service), choisissez la stratégie de planification du service.

      - Pour que le planificateur déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster, choisissez Daemon (Démon).
      - Pour que. le planificateur place et gère le nombre de tâches souhaité dans votre cluster, choisissez Replica.

   b. Si vous choisissez Replica pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

c. Si vous avez choisi Replica, pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage des services entre zones de disponibilité, sélectionnez Rééquilibrage des services entre zones de disponibilité.

d. Pour Période de grâce pour la surveillance de l'état, saisissez la durée (en secondes) pendant laquelle le planificateur de service ignore les surveillances de l'état de Elastic Load Balancing, VPC Lattice et des conteneurs qui ne sont pas saines après le démarrage initial d'une tâche. Si vous ne spécifiez aucune valeur pour la période de grâce de surveillance de l'état, la valeur par défaut 0 est utilisée.

e. Déterminez le type de déploiement de votre service. Développez Options de déploiement, puis spécifiez les paramètres suivants.

Type de déploiement	Étapes	
La mise à jour propagée	a. Pour Min running tasks (Minimum de tâches en cours d'exécuti on), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configur ation de déploiement). b. Pour Max running tasks (Maximum de tâches en cours d'exécuti on), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).	

f. En dessous Configuration du déploiement, pour terminer le déploiement avant que la révision du service cible n'atteigne le nombre total souhaité, configurez des critères de réussite précoce. Vous pouvez également choisir le moment où Amazon ECS nettoie les tâches de la révision du service source.

   i. Allumer Critères de réussite précoces.

   ii. Pour Pourcentage en bonne santé, entrez le pourcentage de tâches qui doivent être en cours d'exécution et en bon état dans la révision du service cible avant qu'Amazon ECS ne termine le déploiement.

   iii. Pour Nettoyage des révisions du service source, choisissez Blocage ou Différé.

g. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

   i. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

      Pour que le logiciel restaure automatiquement le dernier état de déploiement terminé lorsque le disjoncteur de déploiement définit le déploiement comme ayant échoué, sélectionnez Restauration en cas d’échec.

   ii. Pour arrêter un déploiement en fonction des statistiques de l'application, sélectionnez Utiliser CloudWatch alarme (s). Puis, à partir de CloudWatch nom de l'alarme, choisissez les alarmes. Pour créer une nouvelle alarme, accédez au CloudWatch console.

      Pour que le logiciel ramène automatiquement le déploiement au dernier état de déploiement terminé lorsqu'un CloudWatch l'alarme met le déploiement en état d'échec, sélectionnez Annulation en cas de panne.

8. Si votre définition de tâche utilise le mode réseau `awsvpc`, vous pouvez spécifier une configuration réseau personnalisée ; développez Mise en réseau, puis procédez comme suit :

   a. Pour VPC, sélectionnez le VPC à utiliser.

   b. Pour Subnets (Sous-réseaux), sélectionnez un ou plusieurs sous-réseaux du VPC que le planificateur de tâches prend en compte lors du placement de vos tâches.

   c. Pour Security group (Groupe de sécurité), vous pouvez sélectionner un groupe de sécurité existant ou en créer un nouveau. Pour utiliser un groupe de sécurité existant, sélectionnez-le

et passez à l'étape suivante. Pour créer un nouveau groupe de sécurité, sélectionnez Create a new security group (Créer un nouveau groupe de sécurité). Vous devez spécifier un nom de groupe de sécurité et une description, puis ajouter une ou plusieurs règles entrantes pour le groupe de sécurité.

d. Pour Public IP (Adresse IP publique), indiquez si vous voulez attribuer automatiquement une adresse IP publique à l'interface réseau Elastic (ENI) de la tâche.

   AWS Fargate les tâches peuvent se voir attribuer une adresse IP publique lorsqu'elles sont exécutées dans un sous-réseau public afin qu'elles disposent d'un itinéraire vers Internet. Il n’est pas possible d’attribuer une adresse IP publique aux tâches EC2 à l’aide de ce champ. Pour plus d’informations, consultez les sections Options de mise en réseau des tâches Amazon ECS pour Fargate et Allocation d’une interface réseau pour une tâche Amazon ECS.

9. (Facultatif) Pour interconnecter votre service à l’aide de Service Connect, développez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter à d’autres services de l’espace de noms, sélectionnez Côté client uniquement.
      - Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

   c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service. Il peut s'agir d'un espace de noms créé séparément dans le même Région AWS dans votre Compte AWS ou un espace de noms dans la même région qui est partagé avec votre compte à l'aide de AWS Resource Access Manager (AWS RAM). Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

   d. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux des conteneurs à CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens.

Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

Voici une description plus détaillée de chaque destination de journal de conteneur.

- Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
- Amazon Data Firehose : configurez la tâche pour envoyer des journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Elles envoient les journaux vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
- Amazon Kinesis Data Streams : configurez la tâche pour envoyer des journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Elles permettent d'envoyer les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
- Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
- Simple Storage Service (Amazon S3) : configurez la tâche pour envoyer des journaux de conteneur à un compartiment Simple Storage Service (Amazon S3). Les options du pilote de journal par défaut sont définies par défaut, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

e. (Facultatif) Pour activer les journaux d'accès, procédez comme suit :

   i. Élargir Configuration du journal d'accès. Pour Formater, choisissez soit JSON ou TEXT.

   ii. Pour inclure les paramètres de requête dans les journaux d'accès, sélectionnez Inclure les paramètres de requête.

10. (Facultatif) Pour interconnecter votre service à l'aide de la découverte de service, développez Découverte de service, puis spécifiez les éléments suivants :

    a. Sélectionnez Utiliser la découverte de service.

    b. Pour utiliser un nouvel espace de noms, sous Configurer l'espace de noms, choisissez Créer un espace de noms, puis fournissez un nom et une description de l'espace de noms.

Pour utiliser un espace de noms existant, choisissez Sélectionner un espace de noms existant, puis choisissez l'espace de noms que vous souhaitez utiliser.

c. Fournissez des informations sur le service de découverte de service, telles que le nom et la description du service.

d. Pour qu'Amazon ECS effectue des surveillances de l'état périodiques au niveau du conteneur, sélectionnez Activer la propagation de l'état des tâches Amazon ECS.

e. Pour DNS record type (Type de registre DNS), sélectionnez le type de registre DNS à créer pour votre service. La découverte de service Amazon ECS prend uniquement en charge les registres A et SRV, en fonction du mode réseau spécifié par votre définition de tâche. Pour de plus amples informations sur ces types de registres, veuillez consulter Supported DNS Record Types (Types de registres DNS pris en charge) dans le Guide du développeur Amazon Route 53.

- Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `bridge` ou `host`, seuls les enregistrements de type SRV sont pris en charge. Choisissez une combinaison de nom et de port de conteneur à associer au registre.
- Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc`, sélectionnez le type d'enregistrement A ou SRV. Si vous avez choisi A, passez à l'étape suivante. Si vous choisissez SRV, spécifiez le port sur lequel le service est accessible ou une combinaison de nom et de port de conteneur à associer à l'enregistrement.

Pour Durée de vie, saisissez la durée en secondes pendant laquelle un jeu d'enregistrements est conservé en cache par les résolveurs DNS et les navigateurs Web.

11. (Facultatif) Pour interconnecter votre service à l'aide de VPC Lattice, développez Treillis en PVC, puis procédez comme suit :

a. Sélectionnez Activer le VPC Lattice

b. Pour Rôle d'infrastructure, choisissez le rôle d'infrastructure.

Si vous n'avez pas créé de rôle, choisissez Créer un rôle d'infrastructure.

c. Sous Groupes cibles, sélectionnez le ou les groupes cibles. Vous devez choisir au moins un groupe cible et pouvez en avoir jusqu'à cinq. Sélectionnez Ajouter un groupe cible pour ajouter d'autres groupes cibles. Choisissez le Nom du port, le Protocole et le Port pour chaque groupe cible que vous avez choisi.

Pour supprimer un groupe cible, sélectionnez Supprimer.

> **Note**
>
> - Si vous souhaitez ajouter un groupe cible existant, vous devez utiliser l' AWS CLI. Pour obtenir des instructions sur la façon d'ajouter des groupes cibles à l'aide du AWS CLI, voir enregistrez-cibles dans le AWS Command Line Interface Référence.
> - Bien qu'un service VPC Lattice puisse avoir plusieurs groupes cibles, chaque groupe cible ne peut être ajouté qu'à un seul service.

d. Pour terminer la configuration de VPC Lattice, incluez vos nouveaux groupes cibles dans l'action par défaut de l'écouteur ou dans les règles d'un service VPC Lattice existant dans la console VPC Lattice. Pour plus d'informations, consultez la section Règles d'écoute pour votre service VPC Lattice.

12. (Facultatif) Pour configurer un équilibreur de charge pour votre service, développez Load balancing (Répartition de charge).

    Choisissez l'équilibreur de charge.

Pour utiliser cet équilibreur de charge	Faites ceci
Application Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibreur de charge Application Load Balancer ou Use an existing load balancer

Pour utiliser cet équilibreur de charge	Faites ceci	
	(Utiliser un équilibreur de charge existant) pour sélectionner un équilibre ur de charge Application Load Balancer existant. c. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. d. Pour Choose container to load balance (Choisissez le conteneur pour équilibre r la charge), choisissez le conteneur qui héberge le service. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les demandes de connexion. Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. f. Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Application Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes	

Pour utiliser cet équilibreur de charge	Faites ceci	
	vers le premier conteneur défini dans votre définition de tâche. g. Pour Délai d'annulation de l'inscription, saisissez le nombre de secondes dont dispose l'équilibreur de charge pour modifier l'état cible `UNUSED`. La durée par défaut est 300 secondes. h. Pour Health check path (Chemin de surveilla nce de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiqu ement des requêtes pour vérifier l'état de la connexion entre Applicati on Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (`/`).	

Pour utiliser cet équilibreur de charge	Faites ceci
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le conteneur qui héberge le service. d. Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. e. Pour Délai d'annulation de l'inscription, saisissez le nombre de secondes dont dispose l'équilibreur de charge pour modifier l'état cible `UNUSED`. La

Pour utiliser cet équilibreur de charge	Faites ceci	
	durée par défaut est 300 secondes. f. Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Network Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/).	

13. (Facultatif) Pour configurer l’autoscaling du service, développez Autoscaling du service, puis spécifiez les paramètres suivants. Pour utiliser l’autoscaling prédictif, qui examine les données de charge passées à partir des flux de trafic, configurez-le après avoir créé le service. Pour plus d’informations, consultez Utilisation des tendances historiques pour mettre à l’échelle les services Amazon ECS via une mise à l’échelle prédictive.

    a. Pour utiliser la mise à l'échelle automatique du service, sélectionnez Service Auto Scaling (mise à l'échelle automatique du service).

    b. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l’autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

    c. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser par l’autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

    d. Choisissez le type de stratégie. Sous Type de stratégie de mise à l’échelle, choisissez l’une des options suivantes.

Pour utiliser ce type de stratégie	Faites ceci
Suivi des cibles	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible). b. Pour Policy name (Nom de la politique), saisissez un nom de politique. c. Pour Métrique de service ECS, sélectionnez l'une des métriques suivantes. • ECSServiceAverageCPUUtilization — Utilisation moyenne du processeur du service. • ECSServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire du service. • ALBRequestCountPerTarget — Nombre de demandes traitées par cible dans un groupe cible Application Load Balancer. d. PourTarget value (Valeur cible), entrez la valeur conservée par le service pour la métrique sélectionnée.

Pour utiliser ce type de stratégie	Faites ceci	
	e. Pour Scale-out période de refroidissement, entrez le temps, en secondes, qui s'écoule après une activité de scale-out (ajout de tâches) qui doit s'écouler avant qu'une autre activité de scale-out puisse démarrer. f. Pour Scale-in période de refroidissement, entrez le temps, en secondes, qui s'écoule après une activité d'extensi on (suppression de tâches) avant qu'une autre activité d'extension puisse démarrer. g. Pour empêcher la politique d'effectuer une activité de mise à l'échelle horizonta le, sélectionnez Turn off scale-in (Désactiv er la mise à l'échelle horizontale). h. • (Facultatif) Sélection nez Désactiver la mise à l’échelle horizontale si vous souhaitez que votre stratégie de mise à l’échelle augmente	

Pour utiliser ce type de stratégie	Faites ceci	
	horizontalement en cas de hausse du trafic, mais ne doive pas réduire horizontalement lorsque le trafic diminue	

Pour utiliser ce type de stratégie	Faites ceci	
Mise à l'échelle par étapes	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Mise à l'échelle par étapes. b. Pour Nom de la stratégie , saisissez un nom de stratégie. c. Pour Alarm name (Nom de l'alarme), entrez un nom unique pour l'alarme. d. Pour Métrique de service Amazon ECS, sélection nez la métrique à utiliser pour l'alarme. e. Pour Statistique, choisissez la statistique d'alarme. f. Pour Période, choisissez la période de l'alarme. g. Pour Condition d'alarme, choisissez comment comparer la métrique sélectionnée au seuil défini. h. Pour Seuil de comparais on des métriques et Période d'évaluat ion pour déclencher l'alarme, saisissez le seuil utilisé pour l'alarme	

Pour utiliser ce type de stratégie	Faites ceci	
	et la durée d'évaluation du seuil. i. Sous Actions de mise à l'échelle, procédez comme suit : • Pour Action, sélectionnez si vous souhaitez ajouter, supprimer ou définir un nombre spécifique souhaité pour votre service. • Si vous avez choisi d'ajouter ou de supprimer des tâches, pour Valeur, saisissez le nombre de tâches (ou le pourcentage de tâches existante s) à ajouter ou à soustraire lors du lancement de l'action de mise à l'échelle. Si vous avez choisi de spécifier le nombre souhaité, saisissez le nombre de tâches. Pour Type, indiquez si la Valeur est un entier ou un pourcentage du nombre souhaité existant. • Pour Limite inférieure et Limite supérieur	

Pour utiliser ce type de stratégie	Faites ceci	
	e, saisissez les limites inférieure et supérieur e de votre ajustement de mise à l'échelle par étapes. Par défaut, la limite inférieure pour l'ajout d'une politique est le seuil de l'alarme et la limite supérieur e est l'infini positif (+). Par défaut, la limite supérieure pour la suppression d'une politique est le seuil de l'alarme et la limite inférieure est l'infini négatif (-). • (Facultatif) Ajoutez des options de mise à l'échelle supplémen taires. Choisissez Ajouter une nouvelle action de mise à l'échelle, puis répétez les étapes de l'Action de mise à l'échelle. • Pour Temps de stabilisation, saisissez la durée, en secondes, à attendre avant qu'une activité de mise à l'échelle précédente ne prenne effet. Pour	

Pour utiliser ce type de stratégie	Faites ceci	
	une stratégie d'ajout, il s'agit du délai après une activité d'augmentation horizontale pendant lequel la stratégie de mise à l'échelle bloque les activités de réduction horizontale et limite le nombre de tâches pouvant augmenter horizontalement simultanément. Pour une politique de suppression, il s'agit du délai qui doit s'écouler après une activité de réduction horizontale avant qu'une autre activité de réduction puisse commencer.	

14. (Facultatif) Pour utiliser une stratégie de placement des tâches autre que la stratégie par défaut, développez Task Placement (Placement des tâches), puis choisissez parmi les options suivantes.

    Pour plus d'informations, consultez Comment Amazon ECS place les tâches sur des instances de conteneur.

    - Répartition équilibrée par AZ : permet de répartir les tâches entre les zones de disponibilité et les instances de conteneur de la zone de disponibilité.
    - AZ Équilibré BinPack — Répartissez les tâches entre les zones de disponibilité et les instances de conteneurs dont la mémoire disponible est la plus faible.

- BinPack — Répartissez les tâches en fonction de la quantité minimale de processeur ou de mémoire disponible.
- Une tâche par hôte : permet de placer au maximum une tâche du service sur chaque instance de conteneur.
- Personnalisé : permet de définir votre propre stratégie de placement des tâches.

Si vous avez choisi Custom (Personnaliser), définissez l'algorithme de placement des tâches et les règles à prendre en compte lors du placement des tâches.

- Sous Strategy (Stratégie), pour Type et Field (Champ), choisissez l'algorithme et l'entité à utiliser pour l'algorithme.

  Vous pouvez saisir jusqu'à 5 stratégies maximum.
- Sous Contrainte, pour Type et Expression, choisissez la règle et l'attribut pour la contrainte.

  Par exemple, pour définir la contrainte permettant de placer des tâches sur des instances T2, pour Expression, saisissez attribute:ecs.instance-type =~ t2.*.

  Vous pouvez saisir jusqu'à 10 contraintes maximum.

15. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

    Le nom et le type de volume sont configurés lorsque vous créez une révision de définition de tâche et ne peuvent pas être modifiés lors de la création d’un service. Pour mettre à jour le nom et le type du volume, vous devez créer une révision de définition de tâche et créer un service à l’aide de la nouvelle révision.

Pour configurer ce type de volume	Faites ceci
Amazon EBS	a. Pour Type de volume EBS, choisissez le type de volume EBS que vous souhaitez attacher à votre tâche.

Pour configurer ce type de volume	Faites ceci	
	b. Pour Taille (Gio), saisissez une valeur valide pour la taille du volume en gibioctets (Gio). Vous pouvez spécifier une taille de volume minimale de 1 Gio et maximale de 16 384 Gio. Cette valeur est obligatoire sauf si vous fournissez un ID d'instantané. c. Pour IOPS, entrez le nombre maximum de input/output opérations (IOPS) que le volume doit fournir. Cette valeur n'est configurable que pour les types de volume `io1`, `io2` et `gp3`. d. Pour Débit (MiB/s), entrez le débit que le volume doit fournir, en mébioctets par seconde (MiBps, ou MiB/s). Cette valeur n'est configurable que pour le type de volume `gp3`. e. Pour ID d'instantané, choisissez un instantané de volume Amazon EBS existant ou saisissez l'ARN d'un instantané si vous souhaitez	

Pour configurer ce type de volume	Faites ceci	
	créer un volume à partir d'un instantané. Vous pouvez également créer un volume vide en ne choisissant ni ne saisissant aucun ID d'instantané. f. Si vous spécifiez un Identifiant de l'instantané, vous pouvez spécifier Taux d'initialisation du volume (MiB/s). Entrez une valeur comprise entre 100 et 300, dans MiB/s, qui déterminera la vitesse de chargement des données à partir de l'instantané spécifié à l'aide de Identifiant de l'instantané pour la création de volumes. g. Pour Type de système de fichiers, choisissez le type de système de fichiers qui sera utilisé pour le stockage et la récupération des données sur le volume. Vous pouvez choisir le système d'exploitation par défaut ou un type de système de fichiers spécifique. La valeur par défaut pour Linux est XFS. Pour les	

Pour configurer ce type de volume	Faites ceci	
	volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarre pas. h. Pour Rôle d'infrastructure, choisissez un rôle IAM doté des autorisations nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` au rôle, ou vous pouvez utiliser la politique comme guide pour créer et associer votre propre politique avec des autorisations répondant à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, consultez	

Pour configurer ce type de volume	Faites ceci	
	Rôle IAM d'infrastructure Amazon ECS. i. Pour Chiffrement, sélectionnez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si votre compte possède Chiffrement par défaut configuré, le volume sera chiffré avec AWS Key Management Service (AWS KMS) touche spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si tu choisis Personnalisé, vous pouvez spécifier un AWS KMS key de votre choix pour le chiffrement des volumes. Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement par défaut est configuré ou si vous créez un volume à partir d'un instantané chiffré.	

Pour configurer ce type de volume	Faites ceci	
	j. Si vous avez choisi Personnalisé pour Chiffrement, vous devez spécifier AWS KMS key que vous souhaitez utiliser. Pour la clé KMS, choisissez une AWS KMS key ou saisissez un ARN. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétrique gérée par le client, assurez-vous que vous disposez des autorisations appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. k. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume Amazon EBS soit en propageant des balises à partir de la définition de la tâche ou du service, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définition de	

Pour configurer ce type de volume	Faites ceci	
	tâche, choisissez Définitio n de tâche pour Propager des balises à partir de. Si vous souhaitez propager des balises depuis le service, choisissez Service pour Propager des balises à partir de. Si vous choisissez Ne pas propager ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d'informa tions sur la création d'es volumes Amazon EBS, consultez la section Étiquetage d'un volume Amazon EBS.	

16. Pour utiliser ECS Exec afin de déboguer le service, sous Configuration de la résolution des problèmes, sélectionnez Activer ECS Exec.
17. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

Ajoutez ou supprimez une balise.

- [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
  - Pour Clé, saisissez le nom de la clé.
  - Pour Valeur, saisissez la valeur de clé.
- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

18. Choisissez Créer.

Étapes suivantes

Voici les actions supplémentaires à effectuer après avoir créé un service.

- Configurez l’autoscaling prédictif, qui examine les données de charge passées issues des flux de trafic. Pour plus d’informations, consultez Utilisation des tendances historiques pour mettre à l’échelle les services Amazon ECS via une mise à l’échelle prédictive.
- Suivez votre déploiement et consultez l’historique des services pour les services qui utilisent le disjoncteur Amazon ECS. Pour plus d’informations, consultez Affichage de l’historique d’un service à l’aide des déploiements de service Amazon ECS.

Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces

Lorsque vous mettez à jour un service qui utilise la stratégie de déploiement continu, Amazon ECS lance un déploiement qui remplace les tâches de la révision du service source par de nouvelles tâches de la révision du service cible. Amazon ECS termine le déploiement une fois que toutes les conditions suivantes sont remplies :

- La révision du service cible atteint 100 % du nombre souhaité, et toutes ses tâches sont en cours d'exécution et en bon état.

- Le disjoncteur de déploiement ou un CloudWatch l'alarme ne déclenche pas de retour en arrière.
- Si vous utilisez la fonction de restauration par alarme, le temps de cuisson s'écoule.
- Les tâches relatives à la révision du service source sont nettoyées.

Grâce à des critères de réussite précoces, vous pouvez configurer le moment où Amazon ECS termine un déploiement, en fonction de la confiance et des besoins opérationnels de votre charge de travail.

Vous pouvez utiliser des critères de réussite précoce dans les scénarios suivants :

- Vous êtes certain qu'un pourcentage de tâches saines sur la révision du service cible est suffisant pour terminer le déploiement, et vous souhaitez le terminer plus tôt afin que les déploiements suivants, CI/CD les oléoducs et les autres opérations dépendantes peuvent se poursuivre.
- Votre service fonctionne sur des capacités spécialisées ou limitées, telles que GPU-accelerated charges de travail d'inférence, dont le lancement de certaines tâches prend plus de temps en raison de la disponibilité du matériel.
- Votre service comporte des tâches relatives à la révision du service source qui sont lentes à se vider et qui empêcheraient le déploiement de rester ouvert, telles que les tâches associées à des connexions de longue durée.
- Vous voulez le disjoncteur de déploiement ou CloudWatch annulation des alarmes pour protéger le déploiement jusqu'à ce que la révision du service cible atteigne le niveau de santé que vous définissez, et vous ne souhaitez pas de restauration après ce point.

Vous définissez les critères à l'aide des paramètres suivants dans la configuration de déploiement.

Pourcentage en bonne santé

Le `earlySuccessCriteria.healthyPercent` représente le nombre de tâches qui doivent être en cours d'exécution et en bon état sur la révision du service cible avant qu'Amazon ECS ne termine le déploiement, en pourcentage du nombre de tâches souhaité pour le service. Cette valeur est arrondie à la valeur supérieure. Par exemple, si le pourcentage sain est de 90 et que le nombre souhaité est de 100, Amazon ECS termine le déploiement une fois que 90 tâches sont saines. Si le pourcentage sain est de 50 et que le nombre souhaité est de 3, Amazon ECS termine le déploiement une fois que 2 tâches sont saines.

Une tâche est saine lorsqu'elle passe avec succès les tests de santé que vous configurez pour le service.

Amazon ECS lance toutes les tâches restantes relatives à la révision du service cible en dehors du déploiement, par le biais d'une mise à l'échelle régulière du service. Une fois qu'Amazon ECS a terminé le déploiement, le disjoncteur de déploiement et CloudWatch l'annulation de l'alarme ne s'applique plus.

Le pourcentage sain doit se situer entre les services `minimumHealthyPercent` et 100.

Nettoyage des révisions du service source

Le `earlySuccessCriteria.sourceServiceRevisionCleanup` détermine à quel moment Amazon ECS nettoie les tâches depuis la révision du service source. Il comporte deux valeurs, `BLOCKING` et `DEFERRED`.

Avec `BLOCKING`, le déploiement se déroule dans l'ordre suivant :

1. Le pourcentage correct de tâches sont en cours d'exécution et en bon état lors de la révision du service cible.
2. Si vous utilisez la fonction de restauration par alarme, le temps de cuisson s'écoule.
3. Amazon ECS nettoie les tâches relatives à la révision du service source.
4. Amazon ECS termine le déploiement.

Avec `DEFERRED`, le déploiement se déroule dans l'ordre suivant :

1. Le pourcentage correct de tâches sont en cours d'exécution et en bon état lors de la révision du service cible.
2. Si vous utilisez la fonction de restauration par alarme, le temps de cuisson s'écoule.
3. Amazon ECS termine le déploiement.
4. Amazon ECS nettoie les tâches relatives à la révision du service source, en dehors du déploiement.

Avec `DEFERRED`, Amazon ECS essaie de nettoyer les tâches relatives à la révision du service source pendant deux semaines maximum.

Utiliser `DEFERRED` pour les services dotés de connexions de longue durée ou d'une protection évolutive des tâches, où les tâches issues de la révision du service source peuvent devoir continuer à s'exécuter sans maintenir le déploiement ouvert. Ceci est utile lorsque la vidange de la révision

du service source est une opération de longue haleine et que votre CI/CD l'outillage a des limites de temps. Amazon ECS termine le déploiement lorsque le pourcentage normal est atteint et que les tâches relatives à la révision du service source sont épuisées en dehors du déploiement.

Comment Amazon ECS évalue les premiers critères de réussite

Amazon ECS réalise un déploiement qui utilise des critères de réussite précoce une fois que toutes les conditions suivantes sont remplies :

- Le nombre de tâches en cours d'exécution et saines sur la révision de service cible correspond au moins au pourcentage normal du nombre souhaité, arrondi au chiffre supérieur.
- Le disjoncteur de déploiement ou un CloudWatch l'alarme ne déclenche pas de retour en arrière.
- Si vous utilisez la fonction de restauration par alarme, le temps de cuisson s'écoule.
- Si le nettoyage des révisions du service source est `BLOCKING`, les tâches relatives à la révision du service source sont nettoyées.

Amazon ECS lance au moins une tâche sur la révision du service cible et attend que celle-ci soit saine avant d'évaluer le pourcentage de qualité.

Exemple : une tâche saine compte

Nombre souhaité	Pourcentage en bonne santé	Tâches saines requises (arrondies au chiffre supérieur)
2	50	1
3	50	2
10	80	8
100	90	90
10	100	10

Configurer des critères de réussite précoces

Vous pouvez configurer les premiers critères de réussite pour un service nouveau ou existant à l'aide de la console ou du AWS CLI.

Console

Dans le flux de création ou de mise à jour du service, sous Configuration du déploiement, activez les premiers critères de réussite. Entrez une valeur pour Healthy Percent et choisissez une option de nettoyage des révisions du service source, bloquée ou différée.

AWS CLI

Réglez le `earlySuccessCriteria` valeurs dans `--deployment-configuration` paramètre lorsque vous créez ou mettez à jour un service.

```
aws ecs update-service \
    --cluster MyCluster \
    --service MyService \
    --deployment-configuration '{
        "strategy": "ROLLING",
        "earlySuccessCriteria": {
            "enable": true,
            "healthyPercent": 90,
            "sourceServiceRevisionCleanup": "BLOCKING"
        }
    }'
```

Ce que vous observez

- `DescribeServiceDeployments` renvoie l'état du déploiement et les critères de réussite précoce configurés. Pendant que le déploiement est en cours, le décompte des tâches est en temps réel. Une fois le déploiement terminé, le nombre de tâches est un instantané de la date à laquelle le déploiement s'est terminé. Cela s'applique à la fois `BLOCKING` et `DEFERRED`.
- Pour afficher le nombre de tâches en temps réel pour un service, utilisez `DescribeServices`.
- Les déploiements qui se terminent en avance sont inclus lorsque vous filtrez `ListServiceDeployments` par un statut de `SUCCESSFUL`.
- événements de changement d'état de déploiement et AWS CloudTrail les enregistrements indiquent la norme `IN_PROGRESS` pour `SUCCESSFUL` cycle de vie. Le critère de réussite précoce n'ajoute pas de nouveau statut de déploiement.

Considérations

Passez en revue les considérations suivantes avant d'utiliser les premiers critères de réussite.

La surveillance de la restauration s'applique pendant le déploiement

Le disjoncteur de déploiement et CloudWatch les alarmes rollback sont des mécanismes de détection des défaillances pour le déploiement. Ils peuvent annuler le déploiement pendant qu'il est en cours. Une fois qu'Amazon ECS a terminé le déploiement, ces mécanismes n'annulent plus le service, y compris pendant qu'Amazon ECS lance les tâches restantes par le biais d'une mise à l'échelle normale du service. Vous ne pouvez pas arrêter un déploiement une fois celui-ci terminé. Définissez le pourcentage sain sur un pourcentage auquel vous êtes certain que la révision du service cible est saine.

`DEFERRED` nettoyage

Avec `DEFERRED`, Amazon ECS nettoie les tâches relatives à la révision du service source une fois le déploiement terminé. Si les tâches de la révision du service source sont protégées, par une protection intégrée des tâches ou par un autre mécanisme, pendant plus de deux semaines après le déploiement, Amazon ECS ne peut pas les nettoyer. Utiliser `DescribeServices` pour surveiller les tâches relatives à la révision du service source.

Amazon ECS blue/green déploiements

UNE blue/green le déploiement est une méthodologie de publication qui réduit les temps d'arrêt et les risques en exécutant deux environnements de production identiques appelés bleu et vert. Avec Amazon ECS blue/green déploiements, vous pouvez valider les nouvelles révisions de service avant de diriger le trafic de production vers ces derniers. Cette approche fournit un moyen plus sûr de déployer les modifications avec la possibilité de revenir rapidement en arrière si nécessaire.

Avantages

Voici les avantages de l'utilisation blue/green déploiements :

- Réduction des risques en testant le trafic de production avant de passer à la production. Vous pouvez valider le nouveau déploiement avec le trafic de test avant de diriger le trafic de production vers celui-ci.
- Déploiements sans durée d’indisponibilité. L’environnement de production reste disponible tout au long du processus de déploiement, garantissant ainsi la disponibilité continue du service.
- Restauration facile si des problèmes sont détectés. Si des problèmes surviennent avec le déploiement vert, vous pouvez rapidement revenir au déploiement bleu sans interruption de service prolongée.

- Environnement de test contrôlé. L'environnement vert fournit un espace isolé pour tester les nouvelles fonctionnalités avec des modèles de trafic réels avant le déploiement complet.
- Processus de déploiement prévisible. L'approche structurée avec des étapes du cycle de vie définies rend les déploiements plus cohérents et fiables.
- Validation automatisée via des hooks de cycle de vie. Vous pouvez implémenter des tests automatisés à différentes étapes du déploiement afin de vérifier le fonctionnement.

Terminologie

Les éléments suivants sont Amazon ECS blue/green conditions de déploiement :

- Durée de l'intégration : durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le transfert du trafic de production.
- Déploiement bleu : révision actuelle du service de production que vous souhaitez remplacer.
- Déploiement vert : nouvelle version du service que vous souhaitez déployer.
- Étapes du cycle de vie : une série d'événements au cours de l'opération de déploiement, tels que le « après transfert du trafic de production » :
- Hook du cycle de vie : fonction Lambda ou point de pause à une étape spécifique du cycle de vie. Les hooks Lambda invoquent les fonctions Lambda que vous avez définies pour exécuter du code personnalisé. Les crochets de pause interrompent le déploiement et attendent que vous appeliez `ContinueServiceDeployment` pour continuer.
- Listener : ressource Elastic Load Balancing ou VPC Lattice qui vérifie les demandes de connexion à l'aide du protocole et du port que vous configurez. Les règles que vous définissez pour un écouteur déterminent la manière dont Amazon ECS achemine les requêtes vers ses cibles enregistrées.
- Règle : ressource Elastic Load Balancing ou VPC Lattice associée à un écouteur. Une règle définit la manière dont les requêtes sont acheminées et comprend une action, une condition et une priorité.
- Groupe cible : ressource Elastic Load Balancing ou VPC Lattice utilisée pour acheminer les demandes vers une ou plusieurs cibles enregistrées (par exemple, des instances EC2). Lorsque vous créez un écouteur, vous spécifiez un groupe cible pour son action par défaut. Le trafic est transféré vers le groupe cible spécifié dans la règle de l'écouteur.
- Transfert de trafic : processus utilisé par Amazon ECS pour transférer le trafic du déploiement bleu au déploiement vert. Pour les services dotés d'un équilibreur de charge, de Service Connect ou de VPC Lattice, Amazon ECS gère automatiquement le transfert de trafic. Pour les services headless,

Amazon ECS remplace les tâches bleues par des tâches vertes mais ne gère pas le transfert du trafic.

Considérations

Tenez compte des éléments suivants lors du choix d'un type de déploiement :

- Utilisation des ressources : Blue/green les déploiements exécutent temporairement les révisions de service bleues et vertes simultanément, ce qui peut doubler votre utilisation des ressources pendant les déploiements.
- Surveillance du déploiement : Blue/green les déploiements fournissent des informations plus détaillées sur l'état du déploiement, ce qui vous permet de surveiller chaque étape du processus de déploiement.
- Annulation : Blue/green les déploiements facilitent le retour à la version précédente si des problèmes sont détectés, car la révision bleue est maintenue jusqu'à la fin du temps de cuisson.
- Hooks du cycle de vie du Network Load Balancer : si vous utilisez un Network Load Balancer pour blue/green déploiements, il reste 10 minutes supplémentaires pour les étapes du cycle de vie TEST_TRAFFIC_SHIFT et PRODUCTION_TRAFFIC_SHIFT. En effet, Amazon ECS veille à ce que le transfert du trafic soit sécurisé.

Amazon ECS blue/green flux de travail de déploiement de services

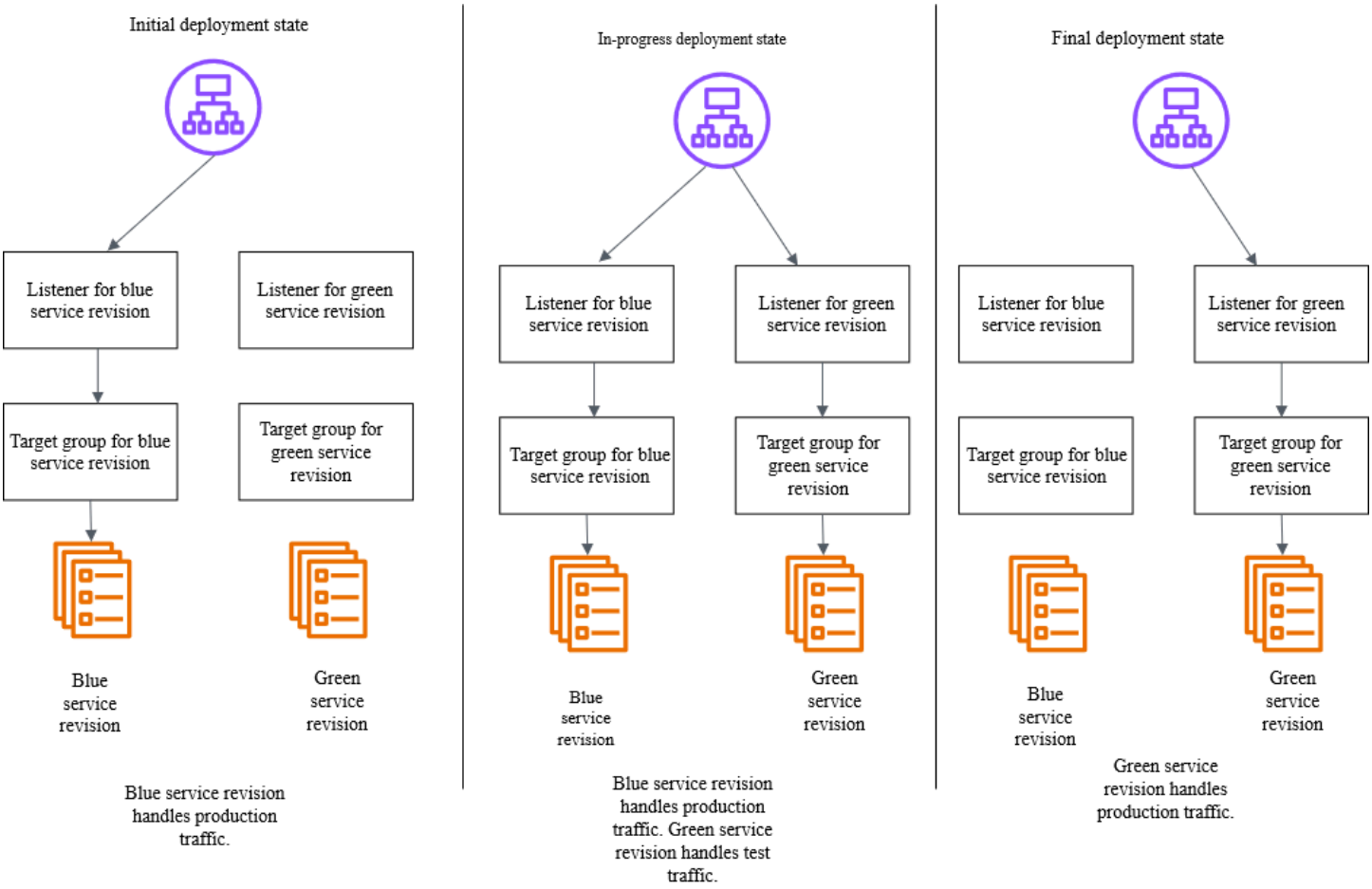
L'Amazon ECS blue/green Le processus de déploiement suit une approche structurée en six phases distinctes qui garantissent des mises à jour sûres et fiables des applications. Chaque phase a un objectif spécifique en validant et en faisant passer votre application de la version actuelle (bleu) à la nouvelle version (vert).

1. Phase de préparation : création de l'environnement vert à côté de l'environnement bleu existant. Cela inclut la mise à disposition de nouvelles révisions de service et la préparation des groupes cibles.
2. Phase de déploiement : déploiement de la nouvelle version du service dans un environnement vert. Amazon ECS lance de nouvelles tâches à l'aide de la révision de service mise à jour tandis que l'environnement bleu continue de desservir le trafic de production.
3. Phase de test : validation de l'environnement vert à l'aide du routage du trafic de test. L'Application Load Balancer dirige les requêtes de test vers l'environnement vert tandis que le trafic de production reste en mode bleu.

4. Phase de transfert du trafic : déplacement du trafic de production du bleu au vert en fonction de votre stratégie de déploiement configurée. Cette phase inclut des points de contrôle de surveillance et de validation.
5. Phase de surveillance : surveillance de l'état de l'application, des métriques de performance et des états d'alarme pendant la durée de l'intégration. Une opération de restauration est lancée lorsque des problèmes sont détectés.
6. Phase d'achèvement : finalisation du déploiement en résiliant l'environnement bleu ou en le maintenant pour d'éventuels scénarios de restauration, en fonction de votre configuration.

Flux de travail

Le schéma suivant illustre l'ensemble blue/green flux de déploiement, illustrant l'interaction entre Amazon ECS et l'équilibreur de charge des applications :



Le processus de déploiement amélioré inclut les étapes détaillées suivantes :

1. État initial : le service bleu (production actuelle) gère 100 % du trafic de production. L'Application Load Balancer possède un écouteur unique avec des règles qui acheminent toutes les requêtes vers le groupe cible bleu contenant des tâches bleues saines.
2. Allocation d'un environnement vert : Amazon ECS crée de nouvelles tâches à l'aide de la définition de tâche mise à jour. Ces tâches sont enregistrées auprès d'un nouveau groupe cible vert, mais ne reçoivent aucun trafic au départ.
3. Validation de la surveillance de l'état : l'Application Load Balancer effectue des surveillances de l'état sur les tâches vertes. Ce n'est que lorsque les tâches vertes ont satisfait aux surveillances de l'état que le déploiement passe à la phase suivante.
4. Routage du trafic de test : si elles sont configurées, les règles d'écoute de l'Application Load Balancer acheminent certains types de trafic (tels que les requêtes avec des en-têtes de test) vers l'environnement vert à des fins de validation, tandis que le trafic de production reste sur l'environnement bleu. Ceci est contrôlé par le même écouteur qui gère le trafic de production, en utilisant des règles différentes basées sur les attributs des requêtes.
5. Transfert du trafic de production : en fonction de la configuration de déploiement, le trafic passe du bleu au vert. Dans ECS blue/green déploiements, il s'agit d'un changement immédiat (en une seule fois) au cours duquel 100 % du trafic passe de l'environnement bleu à l'environnement vert. L'Application Load Balancer utilise un écouteur unique avec des règles d'écoute qui contrôlent la distribution du trafic entre les groupes cibles bleus et verts en fonction des pondérations.
6. Surveillance et validation: Tout au long du transfert de trafic, Amazon ECS surveille CloudWatch les mesures, les états d'alarme et l'état du déploiement. Des déclencheurs de restauration automatique s'activent si des problèmes sont détectés.
7. Durée de l'intégration : durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le transfert du trafic de production.
8. Résiliation de l'environnement bleu : une fois le transfert de trafic et la validation réussis, l'environnement bleu est résilié pour libérer les ressources du cluster, ou maintenu pour permettre une restauration rapide.
9. État final : l'environnement vert devient le nouvel environnement de production, gérant 100 % du trafic. Le déploiement est marqué comme réussi.

Étapes du cycle de vie du déploiement

Le blue/green le processus de déploiement passe par des étapes de cycle de vie distinctes (une série d'événements au cours de l'opération de déploiement, tels que le « transfert du trafic après la production »), chacune étant associée à des responsabilités et à des points de contrôle de validation

spécifiques. La compréhension de ces étapes vous permet de suivre la progression du déploiement et de résoudre les problèmes de manière efficace.

Chaque étape du cycle de vie peut durer jusqu'à 24 heures. Nous recommandons que la valeur reste inférieure à 24 heures. Cela est dû au fait que les processus asynchrones ont besoin de temps pour déclencher les hooks. Le système expire, échoue lors du déploiement, puis lance une restauration après qu'une étape ait atteint 24 heures. CloudFormation les déploiements sont soumis à des restrictions de délai supplémentaires. Tant que la limite de 24 heures reste en vigueur, CloudFormation impose une limite de 36 heures pour l'ensemble du déploiement. CloudFormation fait échouer le déploiement, puis lance une restauration si le processus ne se termine pas dans les 36 heures.

Pour les crochets de pause, vous pouvez configurer le délai d'attente jusqu'à 20 160 minutes (14 jours). Le délai de déploiement global est de 30 jours.

Étapes du cycle de vie	Description	Utiliser cette étape pour le hook de cycle de vie ?
RECONCILE_SERVICE	Cette étape ne se produit que lorsque vous démarrez un nouveau déploiement de service avec plus d'une révision de service dans un état ACTIF.	Oui
PRE_SCALE_UP	La révision de service verte n'a pas commencé. La révision de service bleue gère 100 % du trafic de production. Il n'y a aucun trafic de test.	Oui
SCALE_UP	Le moment où la révision de service verte augmente verticalement jusqu'à 100 % et lance de nouvelles tâches. La révision de service verte ne génère aucun trafic pour le moment.	Non
POST_SCALE_UP	La révision de service verte a commencé. La révision de service bleue gère 100 %	Oui

Étapes du cycle de vie	Description	Utiliser cette étape pour le hook de cycle de vie ?
	du trafic de production. Il n'y a aucun trafic de test.	
TEST_TRAFFIC_SHIFT	Les révisions de service bleues et vertes sont en cours. La révision de service bleue gère 100 % du trafic de production. La révision de service verte passe de 0 % à 100 % du trafic de test.	Oui (Lambda uniquement)
POST_TEST_TRAFFIC_SHIFT	Le transfert du trafic de test est terminé. La révision de service verte gère 100 % du trafic de test.	Oui
MODIFICATION_DE_TRAITEMENT_PRÉALABLE À LA PRODUCTION	Se produit avant le changement de trafic de production. Pour blue/green déploiements, cette étape est invoquée une seule fois.	Oui
PRODUCTION_TRAFFIC_SHIFT	Le trafic de production est transféré vers la révision de service verte. La révision de service verte passe de 0 % à 100 % du trafic de production.	Oui (Lambda uniquement)
POST_PRODUCTION_TRAFFIC_SHIFT	Le transfert du trafic de production est terminé.	Oui
BAKE_TIME	Durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément.	Non
CLEAN_UP	La révision de service bleue a été complètement réduite à 0 tâches en cours d'exécution. Au terme de cette étape, la révision de service verte devient la révision de service de production.	Non

Chaque étape du cycle de vie comprend des points de contrôle de validation intégrés qui doivent être validés avant de passer à l'étape suivante. Si une validation échoue, le déploiement peut être automatiquement annulé pour garantir la disponibilité et la fiabilité du service.

Lorsque vous utilisez une fonction Lambda, celle-ci doit terminer le travail ou renvoyer EN_COURS dans les 15 minutes. Vous pouvez utiliser le `callBackDelaySeconds` pour retarder l'appel à Lambda. Pour plus d'informations, voir fonction app.py dans le fichier sample-amazon-ecs-blue-green-deployment-patterns sur GitHub.

Ressources requises pour Amazon ECS blue/green déploiements

Vous pouvez utiliser Amazon ECS blue/green déploiements avec ou sans transfert de trafic géré. Lorsque votre service utilise Elastic Load Balancing, Service Connect ou VPC Lattice, Amazon ECS gère pour vous le transfert de trafic entre les révisions de service bleues et vertes. Si votre service n'utilise pas d'équilibreur de charge, Service Connect ou VPC Lattice (service headless), vous pouvez toujours utiliser blue/green déploiements pour des déploiements contrôlés, mais Amazon ECS ne gère pas automatiquement le transfert de trafic.

Pour le transfert de trafic géré, configurez l'une des options suivantes :

- Elastic Load Balancing
- Connexion au service
- Treillis en PVC

La liste suivante fournit une vue d'ensemble de haut niveau de ce que vous devez configurer pour Amazon ECS. blue/green déploiements :

- Si votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, Service Connect ou VPC Lattice, configurez les ressources appropriées pour le transfert de trafic géré.
  - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
  - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
  - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.

- VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

  Si votre service est sans interface (pas d'équilibreur de charge, de Service Connect ou de VPC Lattice), vous n'avez pas besoin de configurer des ressources de transfert de trafic supplémentaires.

- Définissez le contrôleur de déploiement de service sur `ECS`.
- Configurez la stratégie de déploiement sur `blue/green` dans votre définition de service.
- Configurez éventuellement des paramètres supplémentaires tels que :
  - La durée de l'intégration pour le nouveau déploiement
  - CloudWatch alarmes pour la restauration automatique
  - Hooks du cycle de vie du déploiement (fonctions Lambda ou crochets de pause qui s'exécutent à des étapes de déploiement spécifiées)

Bonnes pratiques

Suivez ces bonnes pratiques pour réussir Amazon ECS blue/green déploiements :

- Configurez les surveillances de l'état appropriées qui reflètent précisément l'état de votre application.
- Définissez une durée de l'intégration permettant de tester suffisamment le déploiement vert.
- Mettre en œuvre CloudWatch des alarmes pour détecter automatiquement les problèmes et déclencher des annulations.
- Utilisez les hooks de cycle de vie pour effectuer des tests automatisés à chaque étape du déploiement.
- Assurez-vous que votre application peut gérer les révisions de service bleues et vertes exécutées simultanément.
- Prévoyez une capacité de cluster suffisante pour gérer les deux révisions de service pendant le déploiement.
- Testez vos procédures de restauration avant de les mettre en œuvre en production.

Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary

Pour utiliser les équilibreurs de charge des applications avec Amazon ECS blue/green déploiements, vous devez configurer des ressources spécifiques qui permettent l'acheminement du trafic entre les révisions de service bleues et vertes.

Groupes cibles

Pour blue/green pour les déploiements avec Elastic Load Balancing, vous devez créer deux groupes cibles :

- Un groupe cible principal pour la révision de service bleue (trafic de production actuel)
- Un autre groupe cible pour la révision de service verte (nouvelle version)

Les deux groupes cibles doivent être configurés avec les paramètres suivants :

- Type de cible : `IP` (pour Fargate ou EC2 en mode réseau `awsvpc`)
- Protocole : `HTTP` (ou le protocole utilisé par votre application)
- Port : port sur lequel votre application écoute (généralement `80` pour le protocole HTTP)
- VPC : le même VPC que vos tâches Amazon ECS
- Paramètres de surveillance de l'état : configurés pour vérifier correctement l'état de votre application

Au cours d'une blue/green déploiement, Amazon ECS enregistre automatiquement les tâches auprès du groupe cible approprié en fonction de la phase de déploiement.

Example Création de groupes cibles pour un Application Load Balancer

Les commandes CLI suivantes créent deux groupes cibles à utiliser avec un équilibreur de charge d'application dans un blue/green déploiement :

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
```

```
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol HTTP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-path / \
    --health-check-protocol HTTP \
    --health-check-interval-seconds 30 \
    --health-check-timeout-seconds 5 \
    --healthy-threshold-count 2 \
    --unhealthy-threshold-count 2
```

Application Load Balancer

Vous devez créer un Application Load Balancer avec la configuration suivante :

- Schéma : Internet-facing ou interne, selon vos besoins
- Type d'adresse IP : IPv4
- VPC : le même VPC que vos tâches Amazon ECS
- Sous-réseaux : au moins deux sous-réseaux dans des zones de disponibilité différentes
- Groupes de sécurité : groupe de sécurité qui autorise le trafic sur les ports de l'écouteur

Le groupe de sécurité attaché à l'Application Load Balancer doit disposer d'une règle sortante qui autorise le trafic vers le groupe de sécurité associé à vos tâches Amazon ECS.

Example Création d'un équilibreur de charge Application Load Balancer

La commande CLI suivante crée un équilibreur de charge d'application à utiliser dans blue/green déploiement :

```
aws elbv2 create-load-balancer \
    --name my-application-load-balancer \
    --type application \
    --security-groups sg-abcd1234 \
```

```
    --subnets subnet-12345678 subnet-87654321
```

Écouteurs et règles

Pour blue/green déploiements, vous devez configurer les écouteurs sur votre équilibreur de charge d'application :

- Écouteur de production : gère le trafic de production (généralement sur le port 80 ou 443)
  - Transfère initialement le trafic au groupe cible principal (révision de service bleue)
  - Après le déploiement, transfert le trafic vers le groupe cible alternatif (révision de service verte)
- Écouteur de test (facultatif) : gère le trafic de test pour valider la révision de service verte avant de passer au trafic de production.
  - Peut être configuré sur un autre port (par exemple, 8080 ou 8443)
  - Transfère le trafic vers le groupe cible alternatif (révision de service verte) pendant les tests

Au cours d'une blue/green déploiement, Amazon ECS met automatiquement à jour les règles d'écoute pour acheminer le trafic vers le groupe cible approprié en fonction de la phase de déploiement.

Example Création d'un écouteur de production

La commande CLI suivante crée un écouteur de production sur le port 80 qui transmet le trafic au groupe cible principal (bleu) :

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 80 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
blue-target-group/abcdef123456
```

Example Création d'un écouteur de test

La commande CLI suivante crée un écouteur de test sur le port 8080 qui transfère le trafic vers le groupe cible alternatif (vert) :

```
aws elbv2 create-listener \
```

```
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
app/my-application-load-balancer/abcdef123456 \
    --protocol HTTP \
    --port 8080 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Création d'une règle d'écoute pour le routage basé sur le chemin

La commande CLI suivante crée une règle qui transmet le trafic d'un chemin spécifique au groupe cible vert à des fins de test :

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 10 \
    --conditions Field=path-pattern,Values='/test/*' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Example Création d'une règle d'écoute pour le routage basé sur l'en-tête

La commande CLI suivante crée une règle qui transfère le trafic avec un en-tête spécifique vers le groupe cible vert à des fins de test :

```
aws elbv2 create-rule \
    --listener-arn arn:aws:elasticloadbalancing:region:123456789012:listener/app/my-
application-load-balancer/abcdef123456/ghijkl789012 \
    --priority 20 \
    --conditions Field=http-header,HttpHeaderConfig='{Name=X-
Environment,Values=[test]}' \
    --actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
green-target-group/ghijkl789012
```

Configuration du service

Vous devez disposer des autorisations nécessaires pour autoriser Amazon ECS à gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.

Lors de la création ou de la mise à jour d'un service Amazon ECS pour blue/green pour les déploiements avec Elastic Load Balancing, vous devez spécifier la configuration suivante.

Remplacez le *user-input* avec tes valeurs.

Les principaux composants de cette configuration sont les suivants :

- `targetGroupArn` : l'ARN du groupe cible principal (révision de service bleue).
- `alternateTargetGroupArn` : l'ARN du groupe cible alternatif (révision de service verte).
- `productionListenerRule` : l'ARN de la règle d'écoute du trafic de production.
- `roleArn` : l'ARN du rôle qui permet à Amazon ECS de gérer les ressources Elastic Load Balancing.
- `strategy`: Réglez sur `BLUE_GREEN` pour activer blue/green déploiements.
- `bakeTimeInMinutes` : la durée pendant laquelle les révisions de service bleues et vertes se déroulent simultanément après le transfert du trafic de production.
- `TestListenerRule` : l'ARN de la règle d'écoute du trafic de test. Ce paramètre est facultatif.

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/primary-target-group/
abcdef123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:account-id:targetgroup/alternate-target-group/
ghijkl789012",
                "productionListenerRule": "arn:aws:elasticloadbalancing:region:account-
id:listener-rule/app/load-balancer-name/abcdef123456/listener/ghijkl789012/rule/
mnopqr345678",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-elb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
```

```
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flux de trafic pendant le déploiement

Au cours d'une blue/green lors du déploiement avec Elastic Load Balancing, le trafic circule dans le système comme suit :

1. État initial : tout le trafic de production est acheminé vers le groupe cible principal (révision de service bleue).
2. Déploiement de la révision de service verte : Amazon ECS déploie les nouvelles tâches et les enregistre auprès du groupe cible alternatif.
3. Trafic de test : si un écouteur de test est configuré, le trafic de test est acheminé vers le groupe cible alternatif pour valider la révision de service verte.
4. Transfert du trafic de production : Amazon ECS met à jour la règle d'écoute de production pour acheminer le trafic vers le groupe cible alternatif (révision de service verte).
5. Durée de l'intégration : durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le transfert du trafic de production.
6. Achèvement : après un déploiement réussi, la révision de service bleue est résiliée.

Si des problèmes sont détectés lors du déploiement, Amazon ECS peut automatiquement procéder à une restauration en redirigeant le trafic vers le groupe cible principal (révision de service bleue).

Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens

Pour utiliser un équilibreur de charge réseau avec Amazon ECS blue/green déploiements, vous devez configurer des ressources spécifiques qui permettent l'acheminement du trafic entre les révisions de service bleues et vertes. Cette section explique les composants requis et leur configuration.

Lorsque votre configuration inclut un Network Load Balancer, Amazon ECS ajoute un délai de 10 minutes aux étapes du cycle de vie suivantes :

- TEST_TRAFFIC_SHIFT
- PRODUCTION_TRAFFIC_SHIFT

Ce délai est à l'origine des problèmes de synchronisation du Network Load Balancer, qui peuvent entraîner une inadéquation entre les pondérations de trafic configurées et le routage réel du trafic dans le plan de données.

Groupes cibles

Pour blue/green pour les déploiements avec un Network Load Balancer, vous devez créer deux groupes cibles :

- Un groupe cible principal pour la révision de service bleue (trafic de production actuel)
- Un autre groupe cible pour la révision de service verte (nouvelle révision de service)

Les deux groupes cibles doivent être configurés avec les paramètres suivants :

- Type de cible : `ip` (pour Fargate ou EC2 en mode réseau `awsvpc`)
- Protocole : `TCP` (ou le protocole utilisé par votre application)
- Port : port sur lequel votre application écoute (généralement `80` pour le protocole HTTP)
- VPC : le même VPC que vos tâches Amazon ECS
- Paramètres de surveillance de l'état : configurés pour vérifier correctement l'état de votre application

  Pour les surveillances de l'état du protocole TCP, le Network Load Balancer établit une connexion TCP avec la cible. Si la connexion est réussie, la cible est considérée comme saine.

  Pour HTTP/HTTPS tests de santé, le Network Load Balancer envoie un HTTP/HTTPS demande à la cible et vérifie la réponse.

Au cours d'une blue/green déploiement, Amazon ECS enregistre automatiquement les tâches auprès du groupe cible approprié en fonction de la phase de déploiement.

Example Création de groupes cibles pour un Network Load Balancer

Ce qui suit AWS Les commandes CLI créent deux groupes cibles à utiliser avec un équilibreur de charge réseau dans un blue/green déploiement :

```
aws elbv2 create-target-group \
    --name blue-target-group \
    --protocol TCP \
    --port 80 \
```

```
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP

aws elbv2 create-target-group \
    --name green-target-group \
    --protocol TCP \
    --port 80 \
    --vpc-id vpc-abcd1234 \
    --target-type ip \
    --health-check-protocol TCP
```

Network Load Balancer

Vous devez créer un Network Load Balancer avec la configuration suivante :

- Schéma : Internet-facing ou interne, selon vos besoins
- Type d'adresse IP : IPv4
- VPC : le même VPC que vos tâches Amazon ECS
- Sous-réseaux : au moins deux sous-réseaux dans des zones de disponibilité différentes

Contrairement aux Application Load Balancer, les Network Load Balancer fonctionnent au niveau de la couche transport (couche 4) et n'utilisent pas de groupes de sécurité. Vous devez plutôt vous assurer que les groupes de sécurité associés à vos tâches Amazon ECS autorisent le trafic provenant du Network Load Balancer sur les ports de l'écouteur.

Example Créer un Network Load Balancer

Ce qui suit AWS La commande CLI crée un équilibreur de charge réseau à utiliser dans un blue/green déploiement :

```
aws elbv2 create-load-balancer \
    --name my-network-load-balancer \
    --type network \
    --subnets subnet-12345678 subnet-87654321
```

Considérations relatives à l'utilisation de NLB avec blue/green déploiements

Lorsque vous utilisez un équilibreur de charge réseau pour blue/green déploiements, considérez les points suivants :

- Fonctionnement de couche 4 : les Network Load Balancer fonctionnent au niveau de la couche transport (couche 4) et n'inspectent pas le contenu de la couche application (couche 7). Cela signifie que vous ne pouvez pas utiliser les en-têtes ou les chemins HTTP pour les décisions de routage.
- Surveillance de l'état : les surveillances de l'état du Network Load Balancer sont limitées aux protocoles TCP, HTTP ou HTTPS. Pour les surveillances de l'état du protocole TCP, le Network Load Balancer vérifie uniquement que la connexion peut être établie.
- Préservation de la connexion : les Network Load Balancer préservent l'adresse IP source du client, ce qui peut être utile à des fins de sécurité et de journalisation.
- Adresses IP statiques : les Network Load Balancer fournissent des adresses IP statiques pour chaque sous-réseau, ce qui peut être utile pour les listes blanches ou lorsque les clients doivent se connecter à une adresse IP fixe.
- Trafic de test : étant donné que les Network Load Balancer ne prennent pas en charge le routage basé sur le contenu, le trafic de test doit être envoyé vers un port différent du trafic de production.

Écouteurs et règles

Pour blue/green pour les déploiements avec un Network Load Balancer, vous devez configurer les auditeurs :

- Écouteur de production : gère le trafic de production (généralement sur le port 80 ou 443)
  - Transfère initialement le trafic au groupe cible principal (révision de service bleue)
  - Après le déploiement, transfert le trafic vers le groupe cible alternatif (révision de service verte)
- Écouteur de test (facultatif) : gère le trafic de test pour valider la révision de service verte avant de passer au trafic de production.
  - Peut être configuré sur un autre port (par exemple, 8080 ou 8443)
  - Transfère le trafic vers le groupe cible alternatif (révision de service verte) pendant les tests

Contrairement aux Application Load Balancer, les Network Load Balancer ne prennent pas en charge les règles de routage basées sur le contenu. Au lieu de cela, le trafic est acheminé en fonction du port et du protocole de l'écouteur.

Ce qui suit AWS Les commandes CLI créent des écouteurs de production et de test pour un équilibreur de charge réseau :

Remplacez le *`user-input`* avec tes valeurs.

```
aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 80 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-
target-group/1234567890123456

aws elbv2 create-listener \
    --load-balancer-arn arn:aws:elasticloadbalancing:region:123456789012:loadbalancer/
net/my-network-lb/1234567890123456 \
    --protocol TCP \
    --port 8080 \
    --default-actions Type=forward,
 TargetGroupArn=arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-
target-group/1234567890123456
```

Configuration du service

Vous devez disposer des autorisations nécessaires pour autoriser Amazon ECS à gérer les ressources de l'équilibreur de charge dans vos clusters en votre nom. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.

Lors de la création ou de la mise à jour d'un service Amazon ECS pour blue/green pour les déploiements avec un Network Load Balancer, vous devez spécifier la configuration suivante :

Remplacez le *user-input* avec tes valeurs.

Les principaux composants de cette configuration sont les suivants :

- `targetGroupArn` : l'ARN du groupe cible principal (révision de service bleue)
- `alternateTargetGroupArn` : l'ARN du groupe cible alternatif (révision de service verte)
- `productionListenerRule` : l'ARN de l'écouteur du trafic de production.
- `testListenerRule` : (facultatif) l'ARN de l'écouteur du trafic de test
- `roleArn` : l'ARN du rôle qui permet à Amazon ECS de gérer les ressources Network Load Balancer
- `strategy`: Réglez sur `BLUE_GREEN` pour activer blue/green déploiements
- `bakeTimeInMinutes`: durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le déplacement du trafic de production

```
{
    "loadBalancers": [
        {
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/blue-target-
group/1234567890123456",
            "containerName": "container-name",
            "containerPort": 80,
            "advancedConfiguration": {
                "alternateTargetGroupArn":
 "arn:aws:elasticloadbalancing:region:123456789012:targetgroup/green-target-
group/1234567890123456",
                "productionListenerRule":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/1234567890123456",
                "testListenerRule":
 "arn:aws:elasticloadbalancing:region:123456789012:listener/net/my-network-
lb/1234567890123456/2345678901234567",
                "roleArn": "arn:aws:iam::123456789012:role/ecs-nlb-role"
            }
        }
    ],
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Flux de trafic pendant le déploiement

Lors d'une blue/green lors d'un déploiement à l'aide d'un Network Load Balancer, le trafic circule dans le système comme suit :

1. État initial : tout le trafic de production est acheminé vers le groupe cible principal (révision de service bleue).
2. Déploiement de la révision de service verte : Amazon ECS déploie les nouvelles tâches et les enregistre auprès du groupe cible alternatif.
3. Trafic de test : si un écouteur de test est configuré, le trafic de test est acheminé vers le groupe cible alternatif pour valider la révision de service verte.

4. Transfert du trafic de production : Amazon ECS met à jour l'écouteur de production pour acheminer le trafic vers le groupe cible alternatif (révision de service verte).
5. Durée de l'intégration : durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le transfert du trafic de production.
6. Achèvement : après un déploiement réussi, la révision de service bleue est résiliée.

Si des problèmes sont détectés lors du déploiement, Amazon ECS peut automatiquement procéder à une restauration en redirigeant le trafic vers le groupe cible principal (révision de service bleue).

Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary

Lorsque vous utilisez Service Connect avec blue/green déploiements, vous devez configurer des composants spécifiques pour permettre un routage correct du trafic entre les révisions de service bleue et verte. Cette section explique les composants requis et leur configuration.

Présentation de l'architecture

Service Connect développe à la fois des fonctionnalités de découverte de services et de maillage de services par le biais d'un proxy parallèle géré qui est automatiquement injecté dans vos tâches Amazon ECS. Ces proxys gèrent les décisions de routage, les nouvelles tentatives et la collecte de métriques, tandis que AWS Cloud Map fournit le backend du registre des services. Lorsque vous déployez un service avec Service Connect activé, le service s'enregistre dans AWS Cloud Map, et les services clients le découvrent via l'espace de noms.

Dans une implémentation standard de Service Connect, les services client se connectent à des noms de service logiques, et le proxy sidecar gère le routage vers les instances de service réelles. Avec blue/green déploiements, ce modèle est étendu pour inclure le routage du trafic de test via le `testTrafficRules` configuration.

Lors d'une blue/green lors du déploiement, les composants clés suivants fonctionnent ensemble :

- Proxy Service Connect : tout le trafic entre les services passe par le proxy Service Connect, qui prend les décisions de routage en fonction de la configuration.
- AWS Cloud Map Inscription: les déploiements bleus et verts sont enregistrés auprès de AWS Cloud Map, mais le déploiement écologique est initialement enregistré comme point de terminaison « test ».
- Routage du trafic de test : les `testTrafficRules` dans la configuration de Service Connect déterminent comment identifier et acheminer le trafic de test vers le déploiement vert. Cela se fait

par le biais d’un routage basé sur les en-têtes, dans le cadre duquel des en-têtes HTTP spécifiques contenus dans les requêtes dirigent le trafic vers la révision de test. Par défaut, Service Connect reconnaît `x-amzn-ecs-blue-green-test` en-tête pour HTTP-based protocoles lorsqu'aucune règle personnalisée n'est spécifiée.

- Configuration du client : tous les clients de l’espace de noms reçoivent automatiquement les itinéraires de production et de test, mais seules les requêtes correspondant aux règles de test seront transférées vers le déploiement vert.

La puissance de cette approche réside dans le fait qu’elle permet de gérer la complexité de la découverte de service pendant les transitions. À mesure que le trafic passe du déploiement bleu au déploiement vert, tous les mécanismes de connectivité et de découverte sont automatiquement mis à jour. Il n’est pas nécessaire de mettre à jour les enregistrements DNS, de reconfigurer les équilibreurs de charge ou de déployer les modifications apportées à la découverte de service séparément, car le maillage de services gère tout cela.

Routage et tests du trafic

Service Connect fournit des fonctionnalités avancées de routage du trafic pour blue/green déploiements, y compris le routage basé sur les en-têtes et la configuration d'alias client pour les scénarios de test.

Test des règles d’en-tête du trafic

Pendant blue/green déploiements, vous pouvez configurer des règles d'en-tête de trafic de test pour acheminer des demandes spécifiques vers la (nouvelle) révision du service verte à des fins de test. Cela vous permet de valider la nouvelle version avec un trafic contrôlé avant de terminer le déploiement.

Service Connect utilise le routage basé sur les en-têtes pour identifier le trafic de test. Par défaut, Service Connect reconnaît `x-amzn-ecs-blue-green-test` en-tête pour HTTP-based protocoles lorsqu'aucune règle personnalisée n'est spécifiée. Lorsque cet en-tête est présent dans une requête, le proxy Service Connect achemine automatiquement la requête vers le déploiement vert à des fins de test.

Les règles d’en-tête du trafic de test vous permettent :

- d’acheminer les demandes avec des en-têtes spécifiques vers la révision de service verte ;
- de tester de nouvelles fonctionnalités avec un sous-ensemble de trafic ;

- de valider le comportement du service avant le basculement complet du trafic ;
- de mettre en œuvre des stratégies de test canary.
- de réaliser des tests d'intégration dans un environnement similaire à celui de la production.

Le mécanisme de routage basé sur les en-têtes fonctionne en toute transparence avec votre architecture applicative existante. Les services à la clientèle n'ont pas besoin d'être au courant des blue/green processus de déploiement : ils incluent simplement les en-têtes appropriés lors de l'envoi de demandes de test, et le proxy Service Connect gère automatiquement la logique de routage.

> **Important**
>
> Les règles d'en-tête de trafic de test exigent que le mappage des ports de service Service Connect possède `appProtocol` réglé sur `http`, `http2`, ou `grpc`. Si votre service utilise le protocole TCP simple sans protocole de couche application (c'est-à-dire `appProtocol` n'est pas défini), vous ne pouvez pas utiliser la fonction basée sur les en-têtes `testTrafficRules` pour blue/green déploiements. Dans ce cas, utilisez plutôt le routage du trafic de test basé sur les ports.

Pour plus d'informations sur la configuration des règles d'en-tête de trafic de test, voir ServiceConnectTestTrafficHeaderRules dans le Référence de l'API Amazon Elastic Container Service.

Règles de correspondance des en-têtes

Les règles de correspondance des en-têtes définissent les critères d'acheminement du trafic de test pendant blue/green déploiements. Vous pouvez configurer plusieurs conditions de correspondance pour contrôler avec précision les demandes qui sont acheminées vers la révision de service verte.

La correspondance des en-têtes prend en charge :

- la correspondance exacte des valeurs d'en-tête ;
- la vérification de la présence d'en-têtes ;
- Pattern-based appariement
- plusieurs combinaisons d'en-têtes.

Les exemples de cas d'utilisation incluent le routage des demandes avec des chaînes d'agent utilisateur spécifiques, des versions d'API ou des indicateurs de fonctionnalité vers le service vert à des fins de test.

Pour plus d'informations sur la configuration de la correspondance des en-têtes, voir ServiceConnectTestTrafficHeaderMatchRules dans le Référence de l'API Amazon Elastic Container Service.

Alias de client pour blue/green déploiements

Les alias clients fournissent des points de terminaison DNS stables pour les services pendant blue/green déploiements. Ils permettent un routage fluide du trafic entre les révisions de service bleues et vertes sans obliger les applications clientes à modifier leurs points de terminaison de connexion.

Au cours d'une blue/green déploiement, alias du client :

- maintiennent des noms DNS cohérents pour les connexions client ;
- permettent la commutation automatique du trafic entre les révisions de service ;
- prennent en charge les stratégies de migration progressive du trafic ;
- fournissent des fonctionnalités de restauration en redirigeant le trafic vers la version bleue.

Vous pouvez configurer plusieurs alias de client pour différents ports ou protocoles, ce qui permet à des architectures de service complexes de maintenir la connectivité pendant les déploiements.

Pour plus d'informations sur la configuration des alias clients, consultez ServiceConnectClientAlias dans le Référence de l'API Amazon Elastic Container Service.

Pratiques exemplaires en matière de test de routage du trafic :

Lors de la mise en œuvre du routage du trafic pour blue/green pour les déploiements avec Service Connect, tenez compte des meilleures pratiques suivantes :

- Commencer par des tests basés sur les en-têtes : utilisez les règles d'en-tête du trafic de test pour valider le service vert avec un trafic maîtrisé avant de transférer tout le trafic.
- Configurer les surveillances de l'état : assurez-vous que les services bleu et vert disposent de surveillances de l'état appropriés configurés pour empêcher le routage du trafic vers des instances défectueuses.
- Surveiller les métriques de service : suivez les indicateurs de performance clés des deux révisions de service pendant le déploiement afin d'identifier les problèmes à un stade précoce.

- Planifier une stratégie de restauration : configurez les alias clients et les règles de routage pour permettre une restauration rapide du service bleu si des problèmes sont détectés.
- Tester la logique de correspondance des en-têtes : validez vos règles de correspondance des en-têtes dans un environnement hors production avant de les appliquer à des déploiements de production.

Connexion au service blue/green flux de travail de déploiement

Comprendre comment Service Connect gère blue/green Le processus de déploiement vous aide à implémenter et à dépanner efficacement vos déploiements. Le flux de travail suivant montre comment les différents composants interagissent au cours de chaque phase du déploiement.

Phases de déploiement

Un service Connect blue/green le déploiement se déroule en plusieurs phases distinctes :

1. État initial : le service bleu gère 100 % du trafic de production. Tous les services clients de l'espace de noms se connectent au service bleu via le nom de service logique configuré dans Service Connect.
2. Inscription au service écologique: Lorsque le déploiement écologique démarre, il s'enregistre auprès de AWS Cloud Map en tant que point de terminaison « test ». Le proxy Service Connect des services clients reçoit automatiquement les configurations d'itinéraires de production et de test.
3. Routage du trafic de test : les requêtes contenant les en-têtes de trafic de test (tels que `x-amzn-ecs-blue-green-test`) sont automatiquement acheminées vers le service vert par le proxy Service Connect. Le trafic de production continue d'affluer vers le service bleu.
4. Préparation au transfert du trafic : une fois les tests réussis, le processus de déploiement prépare le transfert du trafic de production. Les services bleus et verts restent enregistrés et sains.
5. Transfert du trafic de production : la configuration de Service Connect est mise à jour pour acheminer le trafic de production vers le service vert. Cela se produit automatiquement sans nécessiter de mises à jour du service client ou de modifications du DNS.
6. Durée de l'intégration : durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le transfert du trafic de production.
7. Désenregistrement de Blue Service: Après un transfert de trafic et une validation réussis, le service bleu est désenregistré de AWS Cloud Map et terminé, achevant ainsi le déploiement.

Comportement du proxy Service Connect

Le proxy Service Connect joue un rôle crucial dans la gestion du trafic pendant blue/green déploiements. Comprendre son comportement vous aide à concevoir des stratégies de test et de déploiement efficaces.

Comportements clés du proxy pendant blue/green déploiements :

- Découverte automatique d'itinéraires: Le proxy découvre automatiquement les itinéraires de production et de test depuis AWS Cloud Map sans avoir à redémarrer l'application ni à modifier la configuration.
- Header-Based Routage: Le proxy examine les en-têtes des demandes entrantes et achemine le trafic vers la révision de service appropriée en fonction des règles de trafic de test configurées.
- Intégration de la surveillance de l’état : le proxy n’achemine le trafic que vers des instances de service saines, excluant automatiquement les tâches défectueuses du groupe de routage.
- Nouvelle tentative et disjoncteur : le proxy intègre une logique de réessai et des fonctionnalités de disjoncteur, améliorant ainsi la résilience lors des déploiements.
- Collecte de métriques : le proxy collecte des métriques détaillées pour les services bleus et verts, permettant une surveillance complète lors des déploiements.

Mise à jour de la découverte de service

L'un des principaux avantages de l'utilisation de Service Connect pour blue/green les déploiements sont la gestion automatique des mises à jour de découverte de services. Traditionnel blue/green les déploiements nécessitent souvent des mises à jour DNS complexes ou une reconfiguration de l'équilibreur de charge, mais Service Connect gère ces modifications de manière transparente.

Au cours d’un déploiement, Service Connect gère les éléments suivants :

- Mises à jour de l’espace de noms : l’espace de noms Service Connect inclut automatiquement les points de terminaison de service bleus et verts, avec des règles de routage appropriées.
- Configuration du client : tous les services clients de l’espace de noms reçoivent automatiquement des informations de routage mises à jour sans nécessiter un redémarrage ni un redéploiement.
- Transition progressive : les mises à jour de la découverte de service se font progressivement et en toute sécurité, sans perturber les requêtes en cours.
- Support de restauration : si une restauration est nécessaire, Service Connect peut rapidement rétablir les configurations de découverte de service afin de rediriger le trafic vers le service bleu.

Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary

Pour utiliser Amazon VPC Lattice avec Amazon ECS blue/greenpour les déploiements linéaires et Canary, vous configurez les ressources suivantes :

- Deux groupes cibles VPC Lattice.
- Un service VPC Lattice avec un écouteur.
- Les règles d'écoute qui acheminent le trafic vers les groupes cibles. Une règle d'écouteur de production est requise et une règle d'écouteur de test est facultative.

Lorsque votre service utilise VPC Lattice, Amazon ECS attend les étapes suivantes du cycle de vie de déploiement afin que l'enregistrement de la cible et les modifications de pondération du trafic puissent se propager au plan de données VPC Lattice :

- `POST_SCALE_UP` — environ 3 minutes, afin que les nouvelles tâches soient opérationnelles avant qu'Amazon ECS n'y transfère le trafic.
- `TEST_TRAFFIC_SHIFT`, `PRODUCTION_TRAFFIC_SHIFT`, et `RECONCILE_SERVICE` — environ 90 secondes après chaque changement de poids. Les déploiements Canary modifient les poids deux fois ; les déploiements linéaires modifient les poids une fois par étape.

Groupes cibles

Pour blue/greenpour les déploiements linéaires, linéaires et Canary avec VPC Lattice, vous devez créer deux groupes cibles :

- Un groupe cible principal pour la révision du service bleu (la version qui dessert actuellement le trafic de production)
- Un autre groupe cible pour la révision de service verte (nouvelle version)

Configurez les deux groupes cibles avec les paramètres suivants :

- Type de cible : `IP` (obligatoire pour les tâches Amazon ECS)
- Protocole : `HTTP` (ou le protocole utilisé par votre application)
- Port : port sur lequel votre application écoute (généralement `80` pour le protocole HTTP)
- Type d'adresse IP : `IPV4` ou `IPV6` (correspond à vos tâches)
- VPC : le même VPC que vos tâches Amazon ECS

- Paramètres de surveillance de l'état : configurés pour vérifier correctement l'état de votre application

Les deux groupes cibles échangent leurs rôles à chaque déploiement. Amazon ECS enregistre les tâches de la nouvelle révision de service auprès du groupe cible qui ne dessert pas le trafic de production.

Les deux groupes cibles doivent appartenir au `ACTIVE` état.

Example Création de groupes cibles pour VPC Lattice

Les commandes CLI suivantes créent deux groupes cibles pour blue/green déploiement :

```
aws vpc-lattice create-target-group \
    --name blue-target-group \
    --type IP \
    --config '{
        "port": 80,
        "protocol": "HTTP",
        "ipAddressType": "IPV4",
        "vpcIdentifier": "vpc-abcd1234",
        "healthCheck": {
            "enabled": true,
            "protocol": "HTTP",
            "path": "/",
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 2,
            "unhealthyThresholdCount": 2
        }
    }'

aws vpc-lattice create-target-group \
    --name green-target-group \
    --type IP \
    --config '{
        "port": 80,
        "protocol": "HTTP",
        "ipAddressType": "IPV4",
        "vpcIdentifier": "vpc-abcd1234",
        "healthCheck": {
            "enabled": true,
```

```
            "protocol": "HTTP",
            "path": "/",
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 2,
            "unhealthyThresholdCount": 2
        }
    }'
```

Service VPC Lattice

Vous avez besoin d'un service VPC Lattice avec un écouteur qui applique les règles décrites dans la section suivante. Les clients accèdent au service VPC Lattice via un réseau de service ou via une association VPC, de la même manière que pour un service Amazon ECS qui n'utilise pas blue/green déploiements. Pour en savoir plus sur ces ressources VPC Lattice, consultez Guide de l'utilisateur d'Amazon VPC Lattice.

Le groupe de sécurité associé à vos tâches Amazon ECS doit autoriser le trafic entrant sur le port de conteneur à partir de la liste de préfixes gérés par VPC Lattice (`com.amazonaws.region.vpc-lattice`, ou `com.amazonaws.region.ipv6.vpc-lattice` pour IPv6). Une liste de préfixes gérée est AWS-ensemble géré de blocs CIDR pour VPC Lattice que vous référencez dans une règle de groupe de sécurité. Pour plus d'informations, voir Contrôlez le trafic vers vos services VPC Lattice à l'aide de groupes de sécurité.

Example Création d'un service VPC Lattice et d'un écouteur

Les commandes CLI suivantes créent un service VPC Lattice et un écouteur HTTP. L'action par défaut de l'auditeur renvoie une valeur fixe `404` réponse, de sorte que seules les règles d'écoute acheminent le trafic vers les groupes cibles :

```
aws vpc-lattice create-service \
    --name my-service

aws vpc-lattice create-listener \
    --service-identifier svc-0123456789abcdef0 \
    --name my-listener \
    --protocol HTTP \
    --port 80 \
    --default-action '{
        "fixedResponse": {
            "statusCode": 404
```

```
        }
    }'
```

Associez le service VPC Lattice à un réseau de services afin que vos clients puissent y accéder. Le réseau de service doit déjà exister et être associé au VPC dans lequel vos clients s'exécutent.

Example Associer le service à un réseau de service

La commande CLI suivante associe le service VPC Lattice à un réseau de services :

```
aws vpc-lattice create-service-network-service-association \
    --service-identifier svc-0123456789abcdef0 \
    --service-network-identifier sn-0123456789abcdef0
```

Écouteurs et règles

Vous configurez les règles d'écoute suivantes sur votre service VPC Lattice :

- Règle d'écoute de production (obligatoire) : achemine le trafic de production.
  - Transfère initialement tout le trafic vers l'un des deux groupes cibles.
  - Après un déploiement réussi, redirige tout le trafic vers l'autre groupe cible.
- Règle d'écoute de test (optionnelle) : achemine le trafic de test, afin que vous puissiez valider la nouvelle révision du service avant que le trafic de production n'y soit transféré.
  - Doit correspondre à un ensemble de demandes différent de celui auquel correspond la règle de l'écouteur de production et doit d'abord être évalué. Par exemple, faites correspondre un en-tête ou un chemin spécifique et attribuez à la règle de test un numéro de priorité inférieur à celui de la règle de production, car VPC Lattice évalue d'abord les numéros de priorité inférieurs.
  - Amazon ECS indique la règle des auditeurs de test lors de la nouvelle révision du service lors de la `TEST_TRAFFIC_SHIFT` étape, avant tout déplacement du trafic de production.
  - Si vous ne configurez pas de règle d'écouteur de test, Amazon ECS ignore le routage du trafic de test.

Les deux règles doivent répondre aux exigences suivantes :

- L'action est `forward`. Amazon ECS rejette une règle comportant un `fixedResponse` action.
- La règle doit être transmise aux deux groupes cibles, avec un poids `100` sur le groupe cible qui dessert actuellement ce trafic et ce poids `0` de l'autre. Les deux groupes cibles sont obligatoires.

- La règle d'écoute de test se trouve sur le même service VPC Lattice que la règle d'écoute de production, et il s'agit d'une règle différente.
- Chaque règle d'écoute ne peut être utilisée que par une seule configuration VPC Lattice sur le service Amazon ECS.

> **Note**
>
> La règle d'écoute de production peut également être la règle par défaut de l'écouteur. Dans ce cas, Amazon ECS met à jour l'action par défaut de l'écouteur au lieu d'une règle nommée. Spécifiez donc l'ARN de l'écouteur à la place d'un ARN de règle. Utilisez cette approche pour les auditeurs qui ne prennent pas en charge les règles, telles que `TLS_PASSTHROUGH` auditeurs. Pour tester le trafic sur un tel service, utilisez la règle par défaut d'un second écouteur sur un port différent.

Amazon ECS déplace le trafic en réécrivant l'action directe de ces règles pendant qu'un déploiement est en cours.

Example Création d'une règle d'écoute de production

La commande CLI suivante crée une règle d'écoute de production qui correspond à tous les chemins et transfère tout le trafic vers le groupe cible bleu :

```
aws vpc-lattice create-rule \
    --service-identifier svc-0123456789abcdef0 \
    --listener-identifier listener-0123456789abcdef0 \
    --name production \
    --priority 100 \
    --match '{
        "httpMatch": {
            "pathMatch": {
                "match": {
                    "prefix": "/"
                }
            }
        }
    }' \
    --action '{
        "forward": {
            "targetGroups": [
```

```
                {
                    "targetGroupIdentifier": "tg-0123456789abcdef0",
                    "weight": 100
                },
                {
                    "targetGroupIdentifier": "tg-0fedcba9876543210",
                    "weight": 0
                }
            ]
        }
    }'
```

Example Création d'une règle d'écouteur de test pour le routage basé sur les en-têtes

La commande CLI suivante crée une règle d'écouteur de test qui fait correspondre les requêtes avec un `X-Environment: test` en-tête. Son numéro de priorité est inférieur à la règle des écouteurs de production, c'est pourquoi VPC Lattice l'évalue en premier :

```
aws vpc-lattice create-rule \
    --service-identifier svc-0123456789abcdef0 \
    --listener-identifier listener-0123456789abcdef0 \
    --name test \
    --priority 10 \
    --match '{
        "httpMatch": {
            "headerMatches": [
                {
                    "name": "X-Environment",
                    "match": {
                        "exact": "test"
                    }
                }
            ]
        }
    }' \
    --action '{
        "forward": {
            "targetGroups": [
                {
                    "targetGroupIdentifier": "tg-0123456789abcdef0",
                    "weight": 100
                },
                {
```

```
                    "targetGroupIdentifier": "tg-0fedcba9876543210",
                    "weight": 0
                }
            ]
        }
    }'
```

Configuration du service

Vous devez disposer des autorisations nécessaires pour autoriser Amazon ECS à gérer les ressources VPC Lattice en votre nom. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.

Lorsque vous créez ou mettez à jour un service Amazon ECS pour blue/green, pour les déploiements linéaires ou Canary avec VPC Lattice, spécifiez la configuration suivante.

Remplacez le *user-input* avec tes valeurs.

Les principaux composants de cette configuration sont les suivants :

- `roleArn`: l'ARN du rôle d'infrastructure qui permet à Amazon ECS de gérer les ressources VPC Lattice.
- `targetGroupArn`: L'ARN d'un groupe cible de la paire.
- `portName`: nom du mappage de port dans votre définition de tâche qui reçoit le trafic.
- `alternateTargetGroupArn`: L'ARN de l'autre groupe cible de la paire.
- `productionListenerRule` : l'ARN de la règle d'écoute du trafic de production.
- `testListenerRule` : l'ARN de la règle d'écoute du trafic de test. Ce paramètre est facultatif.
- `strategy`: Réglez sur `BLUE_GREEN`, `LINEAR`, ou `CANARY`.
- `bakeTimeInMinutes`: durée pendant laquelle les deux révisions de service sont exécutées simultanément une fois que le trafic de production a changé.

```
{
    "vpcLatticeConfigurations": [
        {
            "roleArn": "arn:aws:iam::111122223333:role/
ecsInfrastructureRoleVpcLattice",
            "targetGroupArn": "arn:aws:vpc-
lattice:region:111122223333:targetgroup/tg-0123456789abcdef0",
            "portName": "web",
```

```
            "advancedConfiguration": {
                "alternateTargetGroupArn": "arn:aws:vpc-
lattice:region:111122223333:targetgroup/tg-0fedcba9876543210",
                "productionListenerRule": "arn:aws:vpc-
lattice:region:111122223333:service/svc-0123456789abcdef0/
listener/listener-0123456789abcdef0/rule/rule-0123456789abcdef0",
                "testListenerRule": "arn:aws:vpc-
lattice:region:111122223333:service/svc-0123456789abcdef0/
listener/listener-0123456789abcdef0/rule/rule-0fedcba9876543210"
            }
        }
    ],
    "deploymentController": {
        "type": "ECS"
    },
    "deploymentConfiguration": {
        "strategy": "BLUE_GREEN",
        "maximumPercent": 200,
        "minimumHealthyPercent": 100,
        "bakeTimeInMinutes": 5
    }
}
```

Pour plus d'informations sur la création d'un service Amazon ECS qui utilise VPC Lattice, consultez Créez un service qui utilise VPC Lattice.

Flux de trafic pendant le déploiement

Lors d'un déploiement avec VPC Lattice, le trafic circule dans le système comme suit :

1. État initial: La règle d'écoute de production transmet tout le trafic de production au groupe cible de la révision de service en cours (bleue).
2. Déploiement de révisions de services écologiques: Amazon ECS enregistre les nouvelles tâches auprès du groupe cible qui ne dessert pas le trafic de production (le groupe cible vert), en fonction du poids `0`.
3. Propagation des cibles: Amazon ECS attend que les nouvelles cibles soient saines.
4. Trafic de test: Si une règle d'écouteur de test est configurée, le trafic de test est acheminé vers le groupe cible vert pour valider la révision du service vert.
5. Réorientation du trafic de production: Amazon ECS met à jour les pondérations de la règle des auditeurs de production. UNE blue/green le déploiement déplace tout le trafic en une seule fois. Un déploiement linéaire ou canarien déplace le trafic de manière incrémentielle.

6. Temps de cuisson: durée pendant laquelle les deux révisions de service sont exécutées simultanément une fois que le trafic de production a changé.
7. Achèvement: Une fois le déploiement réussi, Amazon ECS annule l'enregistrement et arrête les tâches bleues. Les deux groupes cibles restent attachés aux règles d'écoute, le groupe cible bleu étant pondéré `0`, prêt pour le prochain déploiement.

Considérations

- Ne modifiez pas les actions avancées: Après la configuration initiale, ne modifiez pas les actions de transfert des règles de production et d'écoute de test en dehors d'Amazon ECS. Si une règle d'écoute est modifiée de manière externe, par exemple si une règle est transmise aux deux groupes cibles ou à aucun d'entre eux au début d'un déploiement, le déploiement échouera.
- Comportement de restauration: lors de l'annulation d'un déploiement, qu'il soit déclenché par Amazon CloudWatch En cas d'alarme, d'un disjoncteur de déploiement ou d'une restauration manuelle, Amazon ECS rétablit les pondérations des règles d'écoute pour rediriger tout le trafic vers la révision de service précédente et arrête les nouvelles tâches.
- Passer d'un déploiement continu: Pour faire passer un service Amazon ECS qui utilise VPC Lattice d'un déploiement continu à un blue/green, déploiement linéaire ou Canary, ajoutez le `advancedConfiguration` bloc pour fournir les autres règles relatives au groupe cible et à l'auditeur. Vous pouvez le faire dans la même mise à jour qui modifie la stratégie, ou dans une mise à jour précédente. Le premier déploiement progressif déplace le trafic vers l'autre groupe cible.
- Configurations multiples: Chaque configuration VPC Lattice du service nécessite sa propre paire de groupes cibles et sa propre règle d'écoute de production. Amazon ECS modifie les règles de chaque configuration indépendamment.
- Comptes: Les groupes cibles, le service et les règles d'écoute de VPC Lattice doivent se trouver dans le même compte que le service Amazon ECS.

Création d'un Amazon ECS blue/green déploiement

En utilisant Amazon ECS blue/green lors de déploiements, vous pouvez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Conditions préalables

Effectuez les opérations suivantes avant de démarrer blue/green déploiement.

1. Configurez les autorisations appropriées.
   - Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.
   - Si vous utilisez VPC Lattice, pour plus d'informations sur les autorisations VPC Lattice, consultez `AmazonECSInfrastructureRolePolicyForVpcLattice`.
   - Pour plus d'informations sur les autorisations Lambda, consultez la sectionAutorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.
2. (Facultatif) Pour le transfert de trafic géré, configurez l'une des ressources suivantes. Si votre service est sans interface (pas d'équilibreur de charge, de Service Connect ou de VPC Lattice), vous pouvez ignorer cette étape. Amazon ECS ne gère pas automatiquement le transfert de trafic pour les services headless.
   - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
   - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
   - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
   - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.
3. Décidez si vous souhaitez exécuter des fonctions Lambda pour les étapes du cycle de vie.
   - PRE_SCALE_UP
   - POST_SCALE_UP
   - TEST_TRAFFIC_SHIFT
   - POST_TEST_TRAFFIC_SHIFT
   - PRODUCTION_TRAFFIC_SHIFT
   - POST_PRODUCTION_TRAFFIC_SHIFT

   Pour plus d'informations sur l'utilisation de Lambda, consultez la section Créer une fonction Lambda avec la console dans le Guide du développeur AWS Lambda .

Procédure

Vous pouvez utiliser la console ou AWS CLI pour créer un Amazon ECS blue/green service.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. La page des détails du cluster s'ouvre. b. Dans l'onglet Services choisissez Create (Créer).
Définition de tâche	a. Sur la page Définitions de tâche, sélectionnez la définition de tâche. b. Dans le menu Déployer, choisissez Créer un service.

La page Créer un service s'affiche.

3. Sous Détails du service, procédez comme suit :

   a. Pour Famille de définition de tâche, choisissez la définition de tâche à utiliser. Ensuite, pour Révision de définition de tâche, saisissez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.

4. Pour exécuter le service dans un cluster existant, pour Cluster existant, sélectionnez le cluster. Pour exécuter le service dans un nouveau cluster, choisissez Créer un cluster
5. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Sous Configuration de calcul, choisissez votre option.

Options de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids.	

Options de calcul	Étapes
	Note Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque Fargate est spécifié, pour Version de plateforme, spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

6. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Type de service, choisissez Replica.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

c. Pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage des services entre zones de disponibilité, sélectionnez Rééquilibrage des services entre zones de disponibilité.

d. Pour Période de grâce pour la surveillance de l'état, saisissez la durée (en secondes) pendant laquelle le planificateur de service ignore les surveillances de l'état de Elastic Load Balancing, VPC Lattice et des conteneurs qui ne sont pas saines après le démarrage initial d'une tâche. Si vous ne spécifiez aucune valeur pour la période de grâce de surveillance de l'état, la valeur par défaut 0 est utilisée.

7. a. Pour Durée de l'intégration, saisissez le nombre de minutes pendant lesquelles les révisions de service bleues et vertes seront exécutées simultanément avant que la révision bleue ne soit résiliée. Cela laisse le temps de procéder à des vérifications et à des tests.

b. (Facultatif) Configurez les hooks du cycle de vie pour qu'ils s'exécutent à des étapes spécifiques du déploiement. En dessous Crochets liés au cycle de vie, choisissez Ajouter.

Pour les crochets Lambda :

- Pour Type de cible, choisissez Lambda.
- Pour Fonction Lambda, saisissez le nom ou l'ARN de la fonction.
- Pour Rôle, sélectionnez le rôle IAM qui dispose de l'autorisation requise pour invoquer la fonction Lambda.
- Pour Étapes du cycle de vie, sélectionnez les étapes dans lesquelles la fonction Lambda doit être exécutée.

Pour les crochets de pause :

- Pour Type de cible, choisissez Pause.
- Pour étapes du cycle de vie, sélectionnez les étapes pendant lesquelles le déploiement doit être suspendu.
- (Facultatif) Pour Délai d'attente, configurez la durée et l'action du délai d'attente.

8. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

   Pour que le logiciel restaure automatiquement le dernier état de déploiement terminé lorsque le disjoncteur de déploiement définit le déploiement comme ayant échoué, sélectionnez Restauration en cas d'échec.

b. Pour arrêter un déploiement en fonction des statistiques de l'application, sélectionnez Utiliser CloudWatch alarme (s). Puis, à partir de CloudWatch nom de l'alarme, choisissez les alarmes. Pour créer une nouvelle alarme, accédez au CloudWatch console.

   Pour que le logiciel ramène automatiquement le déploiement au dernier état de déploiement terminé lorsqu'un CloudWatch l'alarme met le déploiement en état d'échec, sélectionnez Annulation en cas de panne.

9. (Facultatif) Pour interconnecter votre service à l'aide de Service Connect, développez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter à d'autres services de l'espace de noms, sélectionnez Côté client uniquement.
      - Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

   c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service. Il peut s'agir d'un espace de noms créé séparément dans le même Région AWS dans votre Compte AWS ou un espace de noms dans la même région qui est partagé avec votre compte à l'aide de AWS Resource Access Manager (AWS RAM). Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

d. (Facultatif) Configurer les règles d'en-tête de trafic de test pour blue/green déploiements. Sous Routage du trafic de test, spécifiez les éléments suivants :

i. Sélectionnez Activer les règles d'en-tête du trafic de test pour acheminer des requêtes spécifiques vers la révision de service verte pendant les tests.

ii. Pour Règles de correspondance des en-têtes, configurez les critères de routage du trafic de test suivants :

- Nom de l'en-tête : saisissez le nom de l'en-tête HTTP correspondant (par exemple, `X-Test-Version` ou `User-Agent`).
- Type de correspondance : choisissez les critères de correspondance :
  - Correspondance exacte : acheminer les requêtes dont la valeur d'en-tête correspond exactement à la valeur spécifiée
  - En-tête présent : acheminer les requêtes contenant l'en-tête spécifié, quelle que soit sa valeur
  - Correspondance de modèle : acheminer les requêtes dont la valeur d'en-tête correspond à un modèle spécifié
- Valeur d'en-tête (si vous utilisez une correspondance exacte ou une correspondance de modèle) : saisissez la valeur ou le modèle à comparer.

Vous pouvez ajouter plusieurs règles de correspondance d'en-têtes pour créer une logique de routage complexe. Les requêtes correspondant à l'une des règles configurées seront acheminées vers la version verte du service à des fins de test.

iii. Choisissez Ajouter une règle d'en-tête pour configurer des conditions de correspondance d'en-tête supplémentaires.

> **Note**
>
> Les règles d'en-tête du trafic de test vous permettent de valider les nouvelles fonctionnalités avec un trafic contrôlé avant de procéder au déploiement complet. Cela vous permet de tester la révision de service verte avec des requêtes spécifiques (telles que celles provenant d'outils de test internes ou d'utilisateurs bêta) tout en maintenant un flux de trafic normal vers la révision de service bleue.

e. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux des conteneurs à CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Amazon Data Firehose : configurez la tâche pour envoyer des journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Elles envoient les journaux vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Amazon Kinesis Data Streams : configurez la tâche pour envoyer des journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Elles permettent d'envoyer les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
   - Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
   - Simple Storage Service (Amazon S3) : configurez la tâche pour envoyer des journaux de conteneur à un compartiment Simple Storage Service (Amazon S3). Les options du pilote de journal par défaut sont définies par défaut, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

10. (Facultatif) Configurer équilibrage de charge pour blue/green déploiement.

Types Elastic Load Balancing	Étapes	
Application Load Balancer	a. Pour Type d'équilibreur de charge, choisisse	

Types Elastic Load Balancing	Étapes	
	z Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibre ur de charge Application Load Balancer ou Use an existing load balancer (Utiliser un équilibreur de charge existant) pour sélectionner un équilibre ur de charge Application Load Balancer existant. c. Pour Conteneur, choisissez le conteneur qui héberge le service. d. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les demandes de connexion . Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. • Pour Règle de production, saisissez	

Types Elastic Load Balancing	Étapes	
	l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de production. • Pour Règle de test, saisissez l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de test. f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Applicati on Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Applicati on Load Balancer achemine les requêtes.	

Types Elastic Load Balancing	Étapes	
	Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscription, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`. La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (`/`). • Dans Nom de groupe alternatif, saisissez le nom du groupe cible	

Types Elastic Load Balancing	Étapes	
	pour votre révision de service de test (vert).	

Types Elastic Load Balancing	Étapes	
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le conteneur qui héberge le service. d. Pour Écouteur de production, choisissez le port d’écouteur de production et le protocole d’écouteur de production. Il s'agit de l'écouteur pour le trafic de révision de votre service de production. e. Pour Port de l’écouteur de test, choisissez le port de l’écouteur de test et le protocole de l’écouteur de test.	

Types Elastic Load Balancing	Étapes	
	Il s'agit de l'écouteur pour le trafic de révision de votre service de test. f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscription, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`.	

Types Elastic Load Balancing	Étapes	
	La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Dans Nom de groupe alternatif, saisissez le nom du groupe cible pour votre révision de service de test (vert).	

11. (Facultatif) Pour utiliser VPC Lattice pour gérer le transfert de trafic pendant blue/green déploiements, extension Réseau en PVC - facultatif et sélectionnez Utiliser VPC Lattice, puis procédez comme suit :

    a. Pour VPC, choisissez le VPC pour vos ressources VPC Lattice. Il doit s'agir du même VPC que votre service Fargate.

    b. Pour Rôle de l'infrastructure, choisissez le rôle IAM de l'infrastructure Amazon ECS qu'Amazon ECS utilise pour gérer les ressources de votre VPC Lattice, ou choisissez Création d'un nouveau rôle d'infrastructure.

c. Pour Nom du port du conteneur, choisissez le nom du port dans la définition de tâche qui correspond au port de conteneur qui reçoit le trafic de VPC Lattice.

d. Pour Service VPC Lattice, choisissez Créer un nouveau service ou Utiliser un service existant. Pour Nom du service, entrez un nom unique pour le service.

e. Pour auditeur, choisissez Créer un nouvel auditeur ou Utiliser un écouteur existant. Pour Port, entrez un port compris entre 1 et 65535. Pour Protocole, choisissez le protocole (par exemple, HTTP).

f. En dessous Règle relative à l'auditeur de production, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire (maximum 200 caractères). Amazon ECS utilise cette règle pour acheminer le trafic de production.

g. (Facultatif, recommandé) Sous Règle de test pour les auditeurs, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire. Amazon ECS utilise cette règle pour acheminer le trafic de test vers la révision du service vert.

h. Pour Groupes cibles, choisissez Création de deux nouveaux groupes cibles, Créez un groupe cible vert, ou Utiliser deux groupes cibles existants. Pour un nouveau groupe cible, entrez un Nom du groupe cible, choisissez un Protocole, entrez un Port de 1 à 65535, choisissez Protocole de bilan de santé, et entrez un Parcours de contrôle de santé. Pour Nom du groupe cible en vert, entrez un nom ; le groupe cible vert utilise la même configuration que le groupe cible bleu.

12. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

    Ajoutez ou supprimez une balise.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.

- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

13. Cliquez sur Créer.

AWS CLI

1. Créez un fichier nommé `service-definition.json` avec le contenu suivant.

   Remplacez le *`user-input`* avec tes valeurs.

```
{
  "serviceName": "myBlueGreenService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
  "deploymentController": {
    "type": "ECS"
  },
  "deploymentConfiguration": {
    "strategy": "BLUE_GREEN",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "bakeTimeInMinutes": 2,
    "alarms": {
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
```

```
      "enable": true
    },
    "lifecycleHooks": [
      {
        "hookTargetArn": "arn:aws:lambda:us-
west-2:7123456789012:function:checkExample",
        "roleArn": "arn:aws:iam::123456789012:role/ECSLifecycleHookInvoke",
        "lifecycleStages": [
          "PRE_SCALE_UP"
        ],
        "hookDetails": {
          "MY_CONFIG_KEY": "my-config-value"
        }
      }
    ]
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-blue-green-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Exécutez `create-service`.

   Remplacez le *`user-input`* avec tes valeurs.

   ```
   aws ecs create-service --cli-input-json file://service-definition.json
   ```

   Vous pouvez également utiliser l'exemple suivant qui crée un blue/green service de déploiement avec une configuration d'équilibreur de charge :

```
aws ecs create-service \
   --cluster "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster" \
   --service-name "blue-green-example-service" \
   --task-definition "nginxServer:1" \
   --launch-type "FARGATE" \
   --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12345,subnet-67890,subnet-abcdef,subnet-
fedcba],securityGroups=[sg-12345],assignPublicIp=ENABLED}" \
   --desired-count 3 \
   --deployment-controller "type=ECS" \
   --deployment-configuration
 "strategy=BLUE_GREEN,maximumPercent=200,minimumHealthyPercent=100,bakeTimeInMinutes=0"
 \
   --load-balancers "targetGroupArn=arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/MyBGtg1/
abcdef1234567890,containerName=nginx,containerPort=80,advancedConfiguration={alternateTa
west-2:123456789012:targetgroup/
MyBGtg2/0987654321fedcba,productionListenerRule=arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/
MyLB/1234567890abcdef/1234567890abcdef,roleArn=arn:aws:iam::123456789012:role/
ELBManagementRole}"
```

Vous pouvez également utiliser l'exemple suivant qui crée un blue/green service de déploiement avec une configuration VPC Lattice :

```
aws ecs create-service \
   --cluster "arn:aws:ecs:us-west-2:123456789012:cluster/MyCluster" \
   --service-name "blue-green-example-service" \
   --task-definition "nginxServer:1" \
   --launch-type "FARGATE" \
   --network-configuration
 "awsvpcConfiguration={subnets=[subnet-12345,subnet-67890,subnet-abcdef,subnet-
fedcba],securityGroups=[sg-12345],assignPublicIp=ENABLED}" \
   --desired-count 3 \
   --deployment-controller "type=ECS" \
   --deployment-configuration
 "strategy=BLUE_GREEN,maximumPercent=200,minimumHealthyPercent=100,bakeTimeInMinutes=0"
 \
   --vpc-lattice-configurations
 "portName=web,roleArn=arn:aws:iam::111122223333:role/ecs-
vpc-lattice-role,targetGroupArn=arn:aws:vpc-lattice:us-
west-2:111122223333:targetgroup/primary-target-group/
```

```
abcdef123456,advancedConfiguration={alternateTargetGroupArn=arn:aws:vpc-
lattice:us-west-2:111122223333:targetgroup/alternate-target-
group/ghijkl789012,productionListenerRule=arn:aws:vpc-lattice:us-
west-2:111122223333:service/svc-0123456789abcdef0/listener/
listener-0123456789abcdef0/rule/
rule-0123456789abcdef0,testListenerRule=arn:aws:vpc-lattice:us-
west-2:111122223333:service/svc-0123456789abcdef0/listener/
listener-0123456789abcdef0/rule/rule-abcdef01234567890}"
```

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il respecte les blue/green motif :
  - La révision de service verte est créée et augmentée horizontalement
  - Le trafic de test est acheminé vers la révision verte (si elle est configurée)
  - Le trafic de production est transféré vers la révision verte
  - Au terme de la durée de l'intégration, la révision bleue est résiliée

Résolution des problèmes d'Amazon ECS blue/green déploiements

Vous trouverez ci-dessous des solutions aux problèmes courants que vous pouvez rencontrer lors de l'utilisation blue/green déploiements avec Amazon ECS. Blue/green des erreurs de déploiement peuvent survenir au cours des phases suivantes :

- Chemin synchrone : erreurs qui apparaissent immédiatement en réponse aux appels d'API `CreateService` ou `UpdateService`.
- Chemin asynchrone : erreurs qui apparaissent dans le champ `statusReason` de `DescribeServiceDeployments` et entraînent une restauration du déploiement

> **Tip**
>
> Vous pouvez utiliser le Serveur Amazon ECS MCP avec des assistants IA pour surveiller les déploiements et résoudre les problèmes de déploiement à l'aide du langage naturel.

Erreurs de configuration de l’équilibreur de charge

La configuration de l'équilibreur de charge est un élément essentiel de blue/green déploiements dans Amazon ECS. Une configuration correcte des règles d’écoute, des groupes cibles et des types d’équilibreurs de charge est essentielle à la réussite des déploiements. Cette section couvre les problèmes courants de configuration des équilibreurs de charge qui peuvent entraîner blue/green échec des déploiements.

Pour résoudre les problèmes d’équilibreur de charge, il est important de comprendre la relation entre les règles d’écoute et les groupes cibles. Dans un blue/green déploiement :

- La règle d’écoute de production dirige le trafic vers la révision de service actuellement active (bleue)
- La règle d’écoute de test peut être utilisée pour valider la nouvelle révision de service (verte) avant le transfert du trafic de production
- Les groupes cibles sont utilisés pour enregistrer les instances de conteneur à partir de chaque révision de service
- Au cours du déploiement, le trafic passe progressivement de la révision de service bleue à la révision de service verte en ajustant les pondérations des groupes cibles dans les règles d’écoute

Erreurs de configuration des règles d’écoute

Les problèmes suivants sont liés à une configuration incorrecte des règles d'écoute pour blue/green déploiements.

Utilisation d’un ARN d’écouteur Application Load Balancer au lieu d’un ARN de règle d’écoute

Message d’erreur : `productionListenerRule has an invalid ARN format. Must be RuleArn for ALB or ListenerArn for NLB. Got: arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/my-alb/abc123/def456`

Solution : lorsque vous utilisez un Application Load Balancer, vous devez spécifier un ARN de règle d’écoute pour `productionListenerRule` et `testListenerRule`, et non un ARN d’écouteur. Pour les Network Load Balancer, vous devez utiliser l’ARN de l’écouteur.

Pour plus d’informations sur la manière de trouver l’ARN de l’écouteur, consultez la section Écouteurs pour vos Application Load Balancer dans le

Guide de l'utilisateur Application Load Balancer. L'ARN d'une règle a le format *arn:aws:elasticloadbalancing:region:account-id:listener-rule/app/...*.

Utiliser la même règle pour les écouteurs de production et de test

Message d'erreur : `The following rules cannot be used as both production and test listener rules: arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/my-alb/abc123/def456/ghi789`

Solution : vous devez utiliser des règles d'écoute différentes pour le trafic de production et de test. Créez une règle d'écoute distincte pour le trafic de test qui achemine le trafic vers votre groupe cible de test.

Groupe cible non associé aux règles d'écoute

Message d'erreur : `Service deployment rolled back because of invalid networking configuration: Target group arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/myAlternateTG/abc123 is not associated with either productionListenerRule or testListenerRule.`

Solution : le groupe cible principal et le groupe cible secondaire doivent être associés à la règle d'écoute de production ou à la règle d'écoute de test. Mettez à jour la configuration de votre équilibreur de charge pour vous assurer que les deux groupes cibles sont correctement associés à vos règles d'écoute.

Règle d'écoute de test manquante avec un Application Load Balancer

Message d'erreur : `For Application LoadBalancer, testListenerRule is required when productionListenerRule is not associated with both targetGroup and alternateTargetGroup`

Solution : lorsque vous utilisez un Application Load Balancer, si aucun des deux groupes cibles n'est associé à la règle d'écoute de production, vous devez spécifier une règle d'écoute de test. Ajoutez une `testListenerRule` à votre configuration et assurez-vous que les deux groupes cibles sont associés à la règle de production ou à la règle d'écoute de test. Pour plus d'informations, consultez la section Écouteurs pour vos Application Load Balancer du Guide de l'utilisateur Application Load Balancer.

Erreurs de configuration du groupe cible

Les problèmes suivants sont liés à une configuration de groupe cible incorrecte pour blue/green déploiements.

Plusieurs groupes cibles recevant du trafic dans une règle d'écoute

Message d'erreur : `Service deployment rolled back because of invalid networking configuration. productionListenerRule arn:aws:elasticloadbalancing:us-west-2:123456789012:listener-rule/app/my-alb/abc123/def456/ghi789 should have exactly one target group serving traffic but found 2 target groups which are serving traffic`

Solution: Avant de démarrer blue/green déploiement, assurez-vous qu'un seul groupe cible reçoit du trafic (dont le poids n'est pas nul) dans votre règle d'écoute. Mettez à jour la configuration de vos règles d'écoute pour définir le poids à zéro pour tout groupe cible qui ne devrait pas recevoir de trafic.

Groupes cibles en double dans les entrées de l'équilibreur de charge

Message d'erreur : `Duplicate targetGroupArn found: arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/myecs-targetgroup/abc123`

Solution : l'ARN de chaque groupe cible doit être unique pour toutes les entrées de l'équilibreur de charge figurant dans votre définition de service. Passez en revue votre configuration et assurez-vous d'utiliser des groupes cibles différents pour chaque entrée de l'équilibreur de charge.

Groupe cible inattendu dans la règle d'écoute de production

Message d'erreur : `Service deployment rolled back because of invalid networking configuration. Production listener rule is forwarding traffic to unexpected target group arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/random-nlb-tg/abc123. Expected traffic to be forwarded to either targetGroupArn: arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/nlb-targetgroup/def456 or alternateTargetGroupArn: arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/nlb-tg-alternate/ghi789`

Solution : la règle d'écoute de production achemine le trafic vers un groupe cible qui n'est pas spécifié dans votre définition de service. Assurez-vous que la règle d'écoute est configurée pour n'acheminer le trafic que vers les groupes cibles spécifiés dans votre définition de service.

Pour plus d'informations, consultez la section Actions de transfert dans le Guide de l'utilisateur Application Load Balancer.

Erreurs de configuration du type d'équilibreur de charge

Les problèmes suivants sont liés à une configuration incorrecte du type d'équilibreur de charge pour blue/green déploiements.

Combinaison des configurations Classic Load Balancer et Application Load Balancer ou Network Load Balancer

Message d'erreur : `All loadBalancers must be strictly either ELBv1 (defining loadBalancerName) or ELBv2 (defining targetGroupArn)`

> **Note**
>
> Les Classic Load Balancer correspondent à la génération précédente d'équilibreurs de charge Elastic Load Balancing. Nous vous recommandons de passer à un équilibreur de charge de génération actuelle. Pour plus d'informations, consultez la section Migration de votre Classic Load Balancer.

Solution:. Utilisez soit uniquement des Classic Load Balancer, soit des Application Load Balancer et des Network Load Balancer.

Pour les Application Load Balancer et Network Load Balancer, ne spécifiez que le champ `targetGroupArn`.

Utilisation d'une configuration avancée avec un Classic Load Balancer

Message d'erreur : `advancedConfiguration field is not allowed with ELBv1 loadBalancers`

Solution: Configuration avancée pour blue/green les déploiements ne sont pris en charge qu'avec les équilibreurs de charge d'application et les équilibreurs de charge réseau. Si vous utilisez un Classic Load Balancer (spécifié avec `loadBalancerName`), vous ne pouvez pas utiliser le champ `advancedConfiguration`. Passez à un Application Load Balancer ou supprimez le champ `advancedConfiguration`.

Configuration avancée incohérente entre les équilibreurs de charge

Message d'erreur : `Either all or none of the provided loadBalancers must have advancedConfiguration defined`

Solution : si vous utilisez plusieurs équilibreurs de charge, vous devez définir `advancedConfiguration` soit pour tous, soit pour aucun. Mettez à jour votre configuration pour garantir la cohérence entre toutes les entrées de l'équilibreur de charge.

Configuration avancée manquante avec blue/green déploiement

Message d'erreur : `advancedConfiguration field is required for all loadBalancers when using a non-ROLLING deployment strategy`

Solution: Lors de l'utilisation d'un blue/green stratégie de déploiement avec Application Load Balancers, vous devez spécifier `advancedConfiguration` champ pour toutes les entrées de l'équilibreur de charge. Ajoutez l'`advancedConfiguration` nécessaire à la configuration de votre équilibreur de charge.

Problèmes d'autorisations

Les problèmes suivants sont liés à des autorisations insuffisantes pour blue/green déploiements.

Stratégie d'approbation manquante concernant le rôle de l'infrastructure

Message d'erreur : `Service deployment rolled back because of invalid networking configuration. ECS was unable to manage the ELB resources due to missing permissions on ECS Infrastructure Role 'arn:aws:iam::123456789012:role/Admin'.`

Solution : le rôle IAM spécifié pour gérer les ressources de l'équilibreur de charge ne dispose pas de la bonne stratégie d'approbation. Mettez à jour la stratégie d'approbation du rôle afin de permettre au service d'assumer ce rôle. La stratégie d'approbation doit inclure :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
```

```
}
```

Autorisations de lecture manquantes sur le rôle d'équilibreur de charge

Message d'erreur : `service myService failed to describe target health on target-group myTargetGroup with (error User: arn:aws:sts::123456789012:assumed-role/myELBRole/ecs-service-scheduler is not authorized to perform: elasticloadbalancing:DescribeTargetHealth because no identity-based policy allows the elasticloadbalancing:DescribeTargetHealth action)`

Solution : le rôle IAM utilisé pour gérer les ressources de l'équilibreur de charge n'est pas autorisé à lire les informations relatives à l'état de la cible. Ajoutez la politique d'autorisation `elasticloadbalancing:DescribeTargetHealth` au rôle. Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.

Autorisations d'écriture manquantes pour le rôle d'équilibreur de charge

Message d'erreur : `service myService failed to register targets in target-group myTargetGroup with (error User: arn:aws:sts::123456789012:assumed-role/myELBRole/ecs-service-scheduler is not authorized to perform: elasticloadbalancing:RegisterTargets on resource: arn:aws:elasticloadbalancing:us-west-2:123456789012:targetgroup/myTargetGroup/abc123 because no identity-based policy allows the elasticloadbalancing:RegisterTargets action)`

Solution : le rôle IAM utilisé pour gérer les ressources de l'équilibreur de charge n'est pas autorisé à enregistrer des cibles. Ajoutez la politique d'autorisation `elasticloadbalancing:RegisterTargets` au rôle. Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.

Autorisation manquante pour modifier les règles d'écoute

Message d'erreur : `Service deployment rolled back because TEST_TRAFFIC_SHIFT lifecycle hook(s) failed. User: arn:aws:sts::123456789012:assumed-role/myELBRole/ECSNetworkingWithELB is not authorized to perform: elasticloadbalancing:ModifyListener on resource: arn:aws:elasticloadbalancing:us-west-2:123456789012:listener/app/`

`my-alb/abc123/def456 because no identity-based policy allows the elasticloadbalancing:ModifyListener action`

Solution : le rôle IAM utilisé pour gérer les ressources de l'équilibreur de charge n'est pas autorisé à modifier les écouteurs. Ajoutez la politique d'autorisation `elasticloadbalancing:ModifyListener` au rôle. Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.

Pour blue/green déploiements, nous vous recommandons de joindre `AmazonECS-ServiceLinkedRolePolicy` politique gérée pour votre rôle d'infrastructure, qui inclut toutes les autorisations nécessaires pour gérer les ressources de l'équilibreur de charge.

Problèmes de hook de cycle de vie

Les problèmes suivants concernent les hooks du cycle de vie dans blue/green déploiements.

Stratégie d'approbation incorrecte sur le rôle du hook Lambda

Message d'erreur : `Service deployment rolled back because TEST_TRAFFIC_SHIFT lifecycle hook(s) failed. ECS was unable to assume role arn:aws:iam::123456789012:role/Admin`

Solution : le rôle IAM spécifié pour le hook de cycle de vie Lambda ne dispose pas de la bonne stratégie d'approbation. Mettez à jour la stratégie d'approbation du rôle afin de permettre au service d'assumer ce rôle. La stratégie d'approbation doit inclure :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

Le hook Lambda renvoie le statut ÉCHEC

Message d'erreur : `Service deployment rolled back because TEST_TRAFFIC_SHIFT lifecycle hook(s) failed. Lifecycle hook target arn:aws:lambda:us-west-2:123456789012:function:myHook returned FAILED status.`

Solution : la fonction Lambda spécifiée en tant que hook de cycle de vie a renvoyé un statut ÉCHEC. Consultez les journaux des fonctions Lambda sur Amazon CloudWatch journaux pour déterminer la raison de l'échec et mettre à jour la fonction pour gérer correctement l'événement de déploiement.

Autorisation manquante pour invoquer une fonction Lambda

Message d'erreur : `Service deployment rolled back because TEST_TRAFFIC_SHIFT lifecycle hook(s) failed. ECS was unable to invoke hook target arn:aws:lambda:us-west-2:123456789012:function:myHook due to User: arn:aws:sts::123456789012:assumed-role/myLambdaRole/ECS-Lambda-Execution is not authorized to perform: lambda:InvokeFunction on resource: arn:aws:lambda:us-west-2:123456789012:function:myHook because no identity-based policy allows the lambda:InvokeFunction action`

Solution : le rôle IAM utilisé pour le hook de cycle de vie Lambda n'est pas autorisé à appeler la fonction Lambda. Ajoutez l'autorisation `lambda:InvokeFunction` à la politique du rôle pour l'ARN de la fonction Lambda spécifique. Pour plus d'informations sur les autorisations Lambda, consultez la section Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

Délai d'expiration d'une fonction Lambda ou réponse non valide

Message d'erreur : `Service deployment rolled back because TEST_TRAFFIC_SHIFT lifecycle hook(s) failed. ECS was unable to parse the response from arn:aws:lambda:us-west-2:123456789012:function:myHook due to HookStatus must not be null`

Solution : la fonction Lambda a expiré ou a renvoyé une réponse non valide. Assurez-vous que votre fonction Lambda renvoie une réponse valide avec un champ `hookStatus` défini sur `SUCCEEDED` ou `FAILED`. Vérifiez également que le délai d'expiration de la fonction Lambda est défini de manière appropriée pour votre logique de validation. Pour plus d'informations, consultez Hooks de cycle de vie pour les déploiements de service Amazon ECS.

Exemple de réponse Lambda valide :

```
{
  "hookStatus": "SUCCEEDED",
  "reason": "Validation passed"
}
```

TargetType non valide pour le hook de cycle de vie

Message d'erreur : `Invalid targetType. Valid values are: AWS_LAMBDA, PAUSE`

Solution: Vérifiez le `targetType` champ dans la configuration de votre hook de cycle de vie. Les valeurs valides sont `AWS_LAMBDA` et `PAUSE`.

Étape du cycle de vie non valide pour le crochet de pause

Message d'erreur : `Invalid lifecycle stage provided for Pause Hook`

Solution: les crochets de pause ne peuvent être configurés que sur `PRE_SCALE_UP`, `POST_SCALE_UP`, `POST_TEST_TRAFFIC_SHIFT`, `PRE_PRODUCTION_TRAFFIC_SHIFT`, ou `POST_PRODUCTION_TRAFFIC_SHIFT`. Vous ne pouvez pas utiliser `TEST_TRAFFIC_SHIFT` ou `PRODUCTION_TRAFFIC_SHIFT` pour les crochets de pause.

Le crochet de pause a expiré

Message d'erreur : `Service deployment rolled back because POST_TEST_TRAFFIC_SHIFT pause hook timed out.`

Solution: Le crochet de pause a expiré avant un `ContinueServiceDeployment` l'appel a été reçu. Vérifiez que votre flux de travail d'automatisation ou d'approbation appelle `ContinueServiceDeployment` avant l'expiration du délai configuré. Vérifiez le `expiresAt` champ dans `DescribeServiceDeployments` pour voir quand le hook devait expirer. Si vous avez besoin de plus de temps, augmentez `timeoutInMinutes` valeur dans la configuration du crochet de pause (maximum 20 160 minutes/14 jours). Si vous souhaitez que le déploiement se poursuive automatiquement à l'expiration du délai au lieu de revenir en arrière, définissez le `timeoutConfiguration.action` pour `CONTINUE`.

Format HookID non valide

Message d'erreur : `Invalid hookId: hookId must follow the format: ecs-{hookType}-{id}`

Solution: Les `hookId` fourni à `ContinueServiceDeployment` ne correspond pas au format attendu. Récupérez le bon `hookId` en appelant `DescribeServiceDeployments` et en vérifiant le `lifecycleHookDetails` tableau. Ne créez pas les HookIds manuellement.

HookId introuvable ou a expiré

Message d'erreur : `Provided hookId is either completed, unavailable, or not associated with the given Resource ARN`

Solution: Les `hookId` n'est plus valide. Cela peut se produire si le hook a déjà expiré, s'il a déjà été poursuivi ou annulé, si un nouveau déploiement a été déclenché (invalidant les HookID précédents) ou si le HookID appartient à un ARN de déploiement de service différent. Récupérez le courant `hookId` en appelant `DescribeServiceDeployments` sur le déploiement actif.

Déploiement à l'état terminal

Message d'erreur : `Continuing a service deployment is not allowed for service deployment in terminal states.`

Solution: le déploiement est déjà terminé, annulé ou arrêté. Vous ne pouvez qu'appeler `ContinueServiceDeployment` sur un déploiement qui est `IN_PROGRESS` avec un crochet de pause `AWAITING_ACTION` statut. Utiliser `DescribeServiceDeployments` pour vérifier l'état actuel du déploiement.

Déploiement du service introuvable

Message d'erreur : `The service deployment does not exist.`

Solution: Les `serviceDeploymentArn` fourni ne correspond à aucun déploiement de service existant. Utiliser `ListServiceDeployments` pour trouver le bon ARN.

Déploiements linéaires Amazon ECS

Les déploiements linéaires déplacent progressivement le trafic de l'ancienne révision du service vers la nouvelle, par incréments égaux au fil du temps, ce qui vous permet de surveiller chaque étape avant de passer à la suivante. Grâce aux déploiements linéaires d'Amazon ECS, contrôlez le rythme du transfert du trafic et validez les nouvelles révisions de service en fonction de volumes croissants de trafic de production. Cette approche fournit un moyen contrôlé de déployer les modifications avec la possibilité de surveiller les performances à chaque incrément.

Ressources impliquées dans un déploiement linéaire

Les ressources suivantes sont impliquées dans les déploiements linéaires d'Amazon ECS :

- Transfert de trafic : processus utilisé par Amazon ECS pour déplacer le trafic de production. Pour les déploiements linéaires Amazon ECS, le trafic est transféré par incréments de pourcentage égaux avec des temps d'attente configurables entre chaque incrément.
- Pourcentage d'étape : pourcentage de trafic à modifier à chaque incrément au cours d'un déploiement linéaire. Ce champ prend Double comme valeur, et les valeurs valides sont comprises entre 3,0 et 100,0.
- Temps de cuisson par étapes : durée d'attente entre chaque incrément de changement de trafic lors d'un déploiement linéaire. Les valeurs valides sont comprises entre 0 et 1 440 minutes.
- Temps de préparation du déploiement : temps, en minutes, qu'Amazon ECS attend après avoir transféré tout le trafic de production vers la nouvelle révision de service, avant de mettre fin à l'ancienne révision de service. Il s'agit de la durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le déplacement du trafic de production.
- Étapes du cycle de vie : une série d’événements au cours de l’opération de déploiement, tels que le « après transfert du trafic de production » :
- Hook du cycle de vie : fonction Lambda ou point de pause à une étape spécifique du cycle de vie. Les hooks Lambda invoquent les fonctions Lambda que vous avez définies pour exécuter du code personnalisé. Les crochets de pause interrompent le déploiement et attendent que vous appeliez `ContinueServiceDeployment` pour continuer. Hooks configurés pour `PRODUCTION_TRAFFIC_SHIFT` ou `PRE_PRODUCTION_TRAFFIC_SHIFT` sont invoqués à chaque étape du changement de trafic de production.
- Groupe cible : ressource Elastic Load Balancing ou VPC Lattice utilisée pour acheminer les demandes vers une ou plusieurs cibles enregistrées (par exemple, des instances EC2). Lorsque vous créez un écouteur, vous spécifiez un groupe cible pour son action par défaut. Le trafic est transféré vers le groupe cible spécifié dans la règle de l'écouteur.
- Listener : ressource Elastic Load Balancing ou VPC Lattice qui vérifie les demandes de connexion à l'aide du protocole et du port que vous configurez. Les règles que vous définissez pour un écouteur déterminent la manière dont Amazon ECS achemine les requêtes vers ses cibles enregistrées.
- Règle : ressource Elastic Load Balancing ou VPC Lattice associée à un écouteur. Une règle définit la manière dont les requêtes sont acheminées et comprend une action, une condition et une priorité.

Considérations

Tenez compte des éléments suivants lors du choix d’un type de déploiement :

- Utilisation des ressources : les déploiements linéaires exécutent temporairement les révisions de service bleue et verte simultanément, ce qui peut doubler votre utilisation des ressources pendant les déploiements.
- Surveillance du déploiement : les déploiements linéaires fournissent des informations détaillées sur l'état du déploiement, ce qui vous permet de surveiller chaque étape du processus de déploiement et chaque incrément de transfert de trafic.
- Annulation : les déploiements linéaires permettent de revenir plus facilement à la version précédente si des problèmes sont détectés, car la révision bleue est maintenue jusqu'à la fin du temps de cuisson.
- Validation progressive : les déploiements linéaires vous permettent de valider la nouvelle révision en fonction de volumes croissants de trafic de production, ce qui renforce la confiance dans le déploiement.
- Durée du déploiement : les déploiements linéaires prennent plus de temps que les déploiements complets en raison du déplacement progressif du trafic et des temps d'attente entre les étapes.

Comment fonctionne le déploiement linéaire

Le processus de déploiement linéaire d'Amazon ECS suit une approche structurée en six phases distinctes qui garantissent des mises à jour d'applications sûres et fiables. Chaque phase a un objectif spécifique en validant et en faisant passer votre application de la version actuelle (bleu) à la nouvelle version (vert).

1. Phase de préparation : création de l’environnement vert à côté de l’environnement bleu existant.
2. Phase de déploiement : déploiement de la nouvelle version du service dans un environnement vert. Amazon ECS lance de nouvelles tâches à l’aide de la révision de service mise à jour tandis que l’environnement bleu continue de desservir le trafic de production.
3. Phase de test : validation de l’environnement vert à l’aide du routage du trafic de test. Votre ressource de transfert de trafic (un équilibreur de charge d'application, un équilibreur de charge réseau, Service Connect ou VPC Lattice) dirige les demandes de test vers l'environnement vert tandis que le trafic de production reste en bleu.
4. Phase de transfert linéaire du trafic : déplacez progressivement le trafic de production du bleu au vert par incréments de pourcentage égaux en fonction de la stratégie de déploiement que vous avez configurée.

5. Phase de surveillance : surveillance de l’état de l’application, des métriques de performance et des états d’alarme pendant la durée de l’intégration. Une opération de restauration est lancée lorsque des problèmes sont détectés.
6. Phase d'achèvement : finalisez le déploiement en mettant fin à l'environnement bleu.

La phase de transfert linéaire du trafic suit les étapes suivantes :

- Initiale : le déploiement commence avec 100 % du trafic acheminé vers la version bleue (actuelle) du service. La (nouvelle) révision de service verte reçoit du trafic de test mais aucun trafic de production au départ.
- Déplacement progressif du trafic - Le trafic passe progressivement du bleu au vert par paliers de pourcentage égaux. Par exemple, avec une configuration par étapes à 10,0 %, les changements de trafic se produisent comme suit :
  - Étape 1 : 10,0 % au vert, 90,0 % au bleu
  - Étape 2 : 20,0 % au vert, 80,0 % au bleu
  - Étape 3 : 30,0 % au vert, 70,0 % au bleu
  - Et ainsi de suite jusqu'à ce que 100 % atteigne le vert
- Temps de cuisson par étapes : entre chaque incrément de changement de trafic, le déploiement attend une durée configurable (temps de cuisson par étapes) pour permettre de surveiller et de valider les performances de la nouvelle révision en fonction de l'augmentation de la charge de trafic. Notez que le temps de cuisson de la dernière étape est ignoré une fois que le trafic est décalé de 100,0 %.
- Hooks du cycle de vie : des fonctions Lambda facultatives ou des crochets de pause peuvent être configurés à différentes étapes du cycle de vie pendant le déploiement pour effectuer une validation, une surveillance ou une logique personnalisée automatisées. Hooks configurés pour `PRODUCTION_TRAFFIC_SHIFT` ou `PRE_PRODUCTION_TRAFFIC_SHIFT` sont invoqués à chaque étape du changement de trafic de production.

Étapes du cycle de vie du déploiement

Le processus de déploiement linéaire passe par différentes étapes du cycle de vie, chacune comportant des responsabilités et des points de contrôle de validation spécifiques. La compréhension de ces étapes vous permet de suivre la progression du déploiement et de résoudre les problèmes de manière efficace.

Chaque étape du cycle de vie peut durer jusqu'à 24 heures et, en outre, chaque étape de changement de trafic dans PRODUCTION_TRAFFIC_SHIFT peut durer jusqu'à 24 heures. Nous recommandons que la valeur reste inférieure à 24 heures. Cela est dû au fait que les processus asynchrones ont besoin de temps pour déclencher les hooks. Le système expire, échoue lors du déploiement, puis lance une restauration après 24 heures d'une étape.

CloudFormation les déploiements sont soumis à des restrictions de délai supplémentaires. Tant que la limite de 24 heures reste en vigueur, CloudFormation impose une limite de 36 heures pour l'ensemble du déploiement. CloudFormation fait échouer le déploiement, puis lance une restauration si le processus ne se termine pas dans les 36 heures.

Pour les crochets de pause, vous pouvez configurer le délai d'attente jusqu'à 20 160 minutes (14 jours). Le délai de déploiement global est de 30 jours.

Étapes du cycle de vie	Description	Support Lifecycle Hook
RECONCILE _SERVICE	Cette étape ne se produit que lorsque vous démarrez un nouveau déploiement de service avec plus d’une révision de service dans un état ACTIF.	Oui
PRE_SCALE_UP	La révision de service verte n’a pas commencé. La révision de service bleue gère 100 % du trafic de production. Il n’y a aucun trafic de test.	Oui
SCALE_UP	Le moment où la révision de service verte augmente verticalement jusqu’à 100 % et lance de nouvelles tâches. La révision de service verte ne génère aucun trafic pour le moment.	Non
POST_SCALE_UP	La révision de service verte a commencé. La révision de service bleue gère 100 % du trafic de production. Il n’y a aucun trafic de test.	Oui
TEST_TRAF FIC_SHIFT	Les révisions de service bleues et vertes sont en cours. La révision de service bleue gère 100 % du trafic de production. La révision de	Oui (Lambda uniquement)

Étapes du cycle de vie	Description	Support Lifecycle Hook
	service verte passe de 0 % à 100 % du trafic de test.	
POST_TEST_TRAFFIC_SHIFT	Le transfert du trafic de test est terminé. La révision de service verte gère 100 % du trafic de test.	Oui
MODIFICATION_DE_TRAITEMENT_PRÉALABLE À LA PRODUCTION	Se produit avant chaque incrément de changement de trafic de production. Les hooks de cycle de vie configurés pour cette étape sont invoqués à chaque étape du changement de trafic.	Oui
PRODUCTION_TRAFFIC_SHIFT	Le trafic passe progressivement du bleu au vert par incréments de pourcentage égaux jusqu'à ce que le vert reçoive 100 % du trafic. Chaque changement de trafic invoque un hook de cycle de vie avec un délai d'expiration de 24 heures.	Oui (Lambda uniquement)
POST_PRODUCTION_TRAFFIC_SHIFT	Le transfert du trafic de production est terminé.	Oui
BAKE_TIME	Durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément.	Non
CLEAN_UP	La révision de service bleue a été complètement réduite à 0 tâches en cours d’exécution. Au terme de cette étape, la révision de service verte devient la révision de service de production.	Non

Ressources requises pour les déploiements linéaires d'Amazon ECS

Pour utiliser un déploiement linéaire avec transfert de trafic géré, votre service doit utiliser l'une des fonctionnalités suivantes :

- Elastic Load Balancing
- Connexion au service
- Treillis en PVC

La liste suivante fournit une vue d'ensemble de haut niveau de ce que vous devez configurer pour les déploiements linéaires Amazon ECS :

- Votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, Service Connect ou VPC Lattice. Configurez les ressources appropriées.
  - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
  - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
  - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
  - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.
- Définissez le contrôleur de déploiement de service sur `ECS`.
- Configurez la stratégie de déploiement sur `linear` dans votre définition de service.
- Configurez éventuellement des paramètres supplémentaires tels que :
  - La durée de l'intégration pour le nouveau déploiement
  - Le pourcentage de trafic à modifier à chaque incrément.
  - Durée d'attente en minutes entre chaque incrément de changement de trafic.
  - CloudWatch alarmes pour la restauration automatique
  - Hooks du cycle de vie du déploiement (fonctions Lambda ou crochets de pause qui s'exécutent à des étapes de déploiement spécifiées)

Bonnes pratiques

Suivez ces bonnes pratiques pour réussir les déploiements linéaires d'Amazon ECS :

- Assurez-vous que votre application peut gérer les deux révisions de service exécutées simultanément.
- Prévoyez une capacité de cluster suffisante pour gérer les deux révisions de service pendant le déploiement.
- Testez vos procédures de restauration avant de les implémenter en production.
- Configurez les surveillances de l'état appropriées qui reflètent précisément l'état de votre application.
- Définissez un temps de cuisson qui permette de tester suffisamment la nouvelle révision du service.
- Mettre en œuvre CloudWatch des alarmes pour détecter automatiquement les problèmes et déclencher des annulations.
- Choisissez des pourcentages de pas et des temps de cuisson qui équilibrent la vitesse de déploiement et les besoins de validation.
- Utilisez des pourcentages d'étape plus faibles (5 à 10 %) pour les applications critiques afin de minimiser l'exposition aux risques.
- Réglez des temps de cuisson par étapes plus longs pour les applications qui ont besoin de temps pour se réchauffer ou se stabiliser.
- Mettre en œuvre CloudWatch des alarmes pour détecter automatiquement les problèmes et déclencher des annulations à tout niveau de trafic.
- Surveillez de près les indicateurs des applications à chaque changement de trafic afin de détecter rapidement la dégradation des performances.
- Assurez-vous que votre application peut gérer les deux révisions de service exécutées simultanément.
- Testez vos procédures de restauration à différents pourcentages de trafic avant de les implémenter en production.

Création d'un déploiement linéaire Amazon ECS

En utilisant les déploiements linéaires Amazon ECS, vous pouvez modifier progressivement le trafic par incréments égaux sur des intervalles de temps spécifiés. Cela permet une validation contrôlée à chaque étape du processus de déploiement.

Conditions préalables

Effectuez les opérations suivantes avant de démarrer un déploiement linéaire.

1. Configurez les autorisations appropriées.
   - Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.
   - Si vous utilisez VPC Lattice, pour plus d'informations sur les autorisations VPC Lattice, consultez `AmazonECSInfrastructureRolePolicyForVpcLattice`.
   - Pour plus d'informations sur les autorisations Lambda, consultez la section Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.
2. Les déploiements linéaires Amazon ECS nécessitent que votre service utilise l'une des fonctionnalités suivantes : Configurez les ressources appropriées.
   - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
   - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
   - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
   - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

Procédure

Vous pouvez utiliser la console ou AWS CLI pour créer un service de déploiement linéaire Amazon ECS.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes	
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service.	

Pour démarrer un service à partir de	Étapes
	La page des détails du cluster s'ouvre. b. Dans l'onglet Services choisissez Create (Créer).
Définition de tâche	a. Sur la page Définitions de tâche, sélectionnez la définition de tâche. b. Dans le menu Déployer, choisissez Créer un service.

La page Créer un service s'affiche.

3. Sous Détails du service, procédez comme suit :

   a. Pour Famille de définition de tâche, choisissez la définition de tâche à utiliser. Ensuite, pour Révision de définition de tâche, saisissez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.

4. Pour exécuter le service dans un cluster existant, pour Cluster existant, sélectionnez le cluster. Pour exécuter le service dans un nouveau cluster, choisissez Créer un cluster

5. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Sous Configuration de calcul, choisissez votre option.

Options de calcul	Étapes
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie :

Options de calcul	Étapes	
	• Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids. **Note** Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de	

Options de calcul	Étapes
	capacité doit être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque Fargate est spécifié, pour Version de plateforme, spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

6. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Type de service, choisissez Replica.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

   c. Pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage des services entre zones de disponibilité, sélectionnez Rééquilibrage des services entre zones de disponibilité.

   d. Pour Période de grâce pour la surveillance de l'état, saisissez la durée (en secondes) pendant laquelle le planificateur de service ignore les surveillances de l'état de Elastic Load Balancing, VPC Lattice et des conteneurs qui ne sont pas saines après le démarrage initial d'une tâche. Si vous ne spécifiez aucune valeur pour la période de grâce de surveillance de l'état, la valeur par défaut 0 est utilisée.

7. En dessous Configuration du déploiement, configurez les paramètres de déploiement linéaire :

   a. Pour Stratégie de déploiement, choisissez Linéaire.

   b. Pour Pourcentage d'augmentation du trafic, entrez le pourcentage de trafic à modifier à chaque incrément (par exemple, 10 % pour décaler le trafic par incréments de 10 %).

   c. Pour Intervalle entre les incréments, entrez le temps d'attente en minutes entre chaque incrément de changement de trafic.

   d. Pour Temps de cuisson, entrez le nombre de minutes pendant lesquelles les révisions de service bleues et vertes seront exécutées simultanément après le dernier changement de trafic avant la fin de la révision bleue.

   e. (Facultatif) Configurez les hooks du cycle de vie pour qu'ils s'exécutent à des étapes spécifiques du déploiement. En dessous Crochets liés au cycle de vie, choisissez Ajouter.

      Pour les crochets Lambda :

      - Pour Type de cible, choisissez Lambda.
      - Pour Fonction Lambda, saisissez le nom ou l’ARN de la fonction.
      - Pour Rôle, sélectionnez le rôle IAM qui dispose de l’autorisation requise pour invoquer la fonction Lambda.
      - Pour Étapes du cycle de vie, sélectionnez les étapes dans lesquelles la fonction Lambda doit être exécutée.

      Pour les crochets de pause :

      - Pour Type de cible, choisissez Pause.
      - Pour étapes du cycle de vie, sélectionnez les étapes pendant lesquelles le déploiement doit être suspendu.
      - (Facultatif) Pour Délai d'attente, configurez la durée et l'action du délai d'attente.

8. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

   Pour que le logiciel restaure automatiquement le dernier état de déploiement terminé lorsque le disjoncteur de déploiement définit le déploiement comme ayant échoué, sélectionnez Restauration en cas d'échec.

b. Pour arrêter un déploiement en fonction des statistiques de l'application, sélectionnez Utiliser CloudWatch alarme (s). Puis, à partir de CloudWatch nom de l'alarme, choisissez les alarmes. Pour créer une nouvelle alarme, accédez au CloudWatch console.

   Pour que le logiciel ramène automatiquement le déploiement au dernier état de déploiement terminé lorsqu'un CloudWatch l'alarme met le déploiement en état d'échec, sélectionnez Annulation en cas de panne.

9. (Facultatif) Pour interconnecter votre service à l'aide de Service Connect, développez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter à d'autres services de l'espace de noms, sélectionnez Côté client uniquement.
      - Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

   c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service. Il peut s'agir d'un espace de noms créé séparément dans le même Région AWS dans votre Compte AWS ou un espace de noms dans la même région qui est partagé avec votre compte à l'aide de AWS Resource Access Manager (AWS RAM). Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

d. (Facultatif) Configurez les règles d'en-tête de trafic de test pour les déploiements linéaires. Sous Routage du trafic de test, spécifiez les éléments suivants :

i. Sélectionnez Activer les règles d'en-tête du trafic de test pour acheminer des requêtes spécifiques vers la révision de service verte pendant les tests.

ii. Pour Règles de correspondance des en-têtes, configurez les critères de routage du trafic de test suivants :

- Nom de l'en-tête : saisissez le nom de l'en-tête HTTP correspondant (par exemple, `X-Test-Version` ou `User-Agent`).
- Type de correspondance : choisissez les critères de correspondance :
  - Correspondance exacte : acheminer les requêtes dont la valeur d'en-tête correspond exactement à la valeur spécifiée
  - En-tête présent : acheminer les requêtes contenant l'en-tête spécifié, quelle que soit sa valeur
  - Correspondance de modèle : acheminer les requêtes dont la valeur d'en-tête correspond à un modèle spécifié
- Valeur d'en-tête (si vous utilisez une correspondance exacte ou une correspondance de modèle) : saisissez la valeur ou le modèle à comparer.

Vous pouvez ajouter plusieurs règles de correspondance d'en-têtes pour créer une logique de routage complexe. Les requêtes correspondant à l'une des règles configurées seront acheminées vers la version verte du service à des fins de test.

iii. Choisissez Ajouter une règle d'en-tête pour configurer des conditions de correspondance d'en-tête supplémentaires.

> **Note**
>
> Les règles d'en-tête du trafic de test vous permettent de valider les nouvelles fonctionnalités avec un trafic contrôlé avant de procéder au déploiement complet. Cela vous permet de tester la révision de service verte avec des requêtes spécifiques (telles que celles provenant d'outils de test internes ou d'utilisateurs bêta) tout en maintenant un flux de trafic normal vers la révision de service bleue.

e. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux des conteneurs à CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Amazon Data Firehose : configurez la tâche pour envoyer des journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Elles envoient les journaux vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Amazon Kinesis Data Streams : configurez la tâche pour envoyer des journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Elles permettent d'envoyer les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
   - Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
   - Simple Storage Service (Amazon S3) : configurez la tâche pour envoyer des journaux de conteneur à un compartiment Simple Storage Service (Amazon S3). Les options du pilote de journal par défaut sont définies par défaut, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

10. (Facultatif) Configurer équilibrage de charge pour un déploiement linéaire.

Types Elastic Load Balancing	Étapes	
Application Load Balancer	a. Pour Type d'équilibreur de charge, choisisse	

Types Elastic Load Balancing	Étapes	
	z Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibre ur de charge Application Load Balancer ou Use an existing load balancer (Utiliser un équilibreur de charge existant) pour sélectionner un équilibre ur de charge Application Load Balancer existant. c. Pour Conteneur, choisissez le conteneur qui héberge le service. d. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les demandes de connexion . Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. • Pour Règle de production, saisissez	

Types Elastic Load Balancing	Étapes	
	l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de production. • Pour Règle de test, saisissez l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de test. f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Applicati on Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Applicati on Load Balancer achemine les requêtes.	

Types Elastic Load Balancing	Étapes	
	Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscription, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`. La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (`/`). • Pour Autre nom de groupe, entrez le nom du groupe cible pour	

Types Elastic Load Balancing	Étapes	
	la révision de votre service de test.	

Types Elastic Load Balancing	Étapes	
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le conteneur qui héberge le service. d. Pour Écouteur de production, choisissez le port d’écouteur de production et le protocole d’écouteur de production. Il s'agit de l'écouteur pour le trafic de révision de votre service de production. e. Pour Port de l’écouteur de test, choisissez le port de l’écouteur de test et le protocole de l’écouteur de test.	

Types Elastic Load Balancing	Étapes	
	Il s'agit de l'écouteur pour le trafic de révision de votre service de test. f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscription, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`.	

Types Elastic Load Balancing	Étapes	
	La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Dans Nom de groupe alternatif, saisissez le nom du groupe cible pour votre révision de service de test (vert).	

11. (Facultatif) Pour utiliser VPC Lattice pour gérer le transfert de trafic lors de déploiements linéaires, développez Réseau en PVC - facultatif et sélectionnez Utiliser VPC Lattice, puis procédez comme suit :

    a. Pour VPC, choisissez le VPC pour vos ressources VPC Lattice. Il doit s'agir du même VPC que votre service Fargate.

    b. Pour Rôle de l'infrastructure, choisissez le rôle IAM de l'infrastructure Amazon ECS qu'Amazon ECS utilise pour gérer les ressources de votre VPC Lattice, ou choisissez Création d'un nouveau rôle d'infrastructure.

c. Pour Nom du port du conteneur, choisissez le nom du port dans la définition de tâche qui correspond au port de conteneur qui reçoit le trafic de VPC Lattice.

d. Pour Service VPC Lattice, choisissez Créer un nouveau service ou Utiliser un service existant. Pour Nom du service, entrez un nom unique pour le service.

e. Pour auditeur, choisissez Créer un nouvel auditeur ou Utiliser un écouteur existant. Pour Port, entrez un port compris entre 1 et 65535. Pour Protocole, choisissez le protocole (par exemple, HTTP).

f. En dessous Règle relative à l'auditeur de production, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire (maximum 200 caractères). Amazon ECS utilise cette règle pour acheminer le trafic de production.

g. (Facultatif, recommandé) Sous Règle de test pour les auditeurs, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire. Amazon ECS utilise cette règle pour acheminer le trafic de test vers la révision du service vert.

h. Pour Groupes cibles, choisissez Création de deux nouveaux groupes cibles, Créez un groupe cible vert, ou Utiliser deux groupes cibles existants. Pour un nouveau groupe cible, entrez Nom du groupe cible, choisissez un Protocole, entrez un Port de 1 à 65535, choisissez Protocole de bilan de santé, et entrez un Parcours de contrôle de santé. Pour Nom du groupe cible en vert, entrez un nom ; le groupe cible vert utilise la même configuration que le groupe cible bleu.

12. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

    Ajoutez ou supprimez une balise.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.

- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

13. Cliquez sur Créer.

AWS CLI

1. Créez un fichier nommé `linear-service-definition.json` avec le contenu suivant.

   Remplacez le *`user-input`* avec tes valeurs.

```
{
  "serviceName": "myLinearService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
  "deploymentController": {
    "type": "ECS"
  },
  "deploymentConfiguration": {
    "strategy": "LINEAR",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "linearConfiguration": {
      "stepPercentage": 10.0,
      "stepBakeTimeInMinutes":6
    },
    "bakeTimeInMinutes": 10,
    "alarms": {
```

```
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
      "enable": true
    }
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-linear-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Exécutez `create-service`.

```
aws ecs create-service --cli-input-json file://linear-service-definition.json
```

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il suit le schéma linéaire :
  - La révision de service verte est créée et augmentée horizontalement
  - Le trafic est déplacé de manière incrémentielle aux intervalles spécifiés
  - Chaque incrément attend l'intervalle spécifié avant le prochain décalage
  - Une fois que tout le trafic est modifié, le temps de cuisson commence
  - Au terme de la durée de l'intégration, la révision bleue est résiliée

Déploiements canary d’Amazon ECS

Les déploiements Canary acheminent d'abord un petit pourcentage du trafic vers la nouvelle révision pour les tests initiaux, puis transfèrent tout le trafic restant en une seule fois une fois la phase Canary terminée avec succès. Avec les déploiements Amazon ECS Canary, validez les nouvelles révisions de service avec un trafic utilisateur réel tout en minimisant l'exposition aux risques. Cette approche fournit un moyen contrôlé de déployer les modifications avec la possibilité de surveiller les performances et de revenir rapidement en arrière si des problèmes sont détectés.

Ressources impliquées dans le déploiement d'un canari

Les ressources suivantes sont impliquées dans les déploiements d'Amazon ECS Canary :

- Transfert de trafic : processus utilisé par Amazon ECS pour déplacer le trafic de production. Pour les déploiements Amazon ECS Canary, le trafic est transféré en deux phases : d'abord vers le pourcentage Canary, puis pour terminer le déploiement.
- Pourcentage canarien : pourcentage du trafic acheminé vers la nouvelle version pendant la période d'évaluation.
- Durée de cuisson de Canary : durée de surveillance de la version Canary avant de procéder au déploiement complet.
- Temps de préparation du déploiement : temps, en minutes, qu'Amazon ECS attend après avoir transféré tout le trafic de production vers la nouvelle révision de service, avant de mettre fin à l'ancienne révision de service. Il s'agit de la durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément après le déplacement du trafic de production.
- Étapes du cycle de vie : une série d’événements au cours de l’opération de déploiement, tels que le « après transfert du trafic de production » :
- Hook du cycle de vie : fonction Lambda ou point de pause à une étape spécifique du cycle de vie. Les hooks Lambda invoquent les fonctions Lambda que vous avez définies pour exécuter du code personnalisé. Les crochets de pause interrompent le déploiement et attendent que vous appeliez `ContinueServiceDeployment` pour continuer.
- Groupe cible : ressource Elastic Load Balancing ou VPC Lattice utilisée pour acheminer les demandes vers une ou plusieurs cibles enregistrées (par exemple, des instances EC2). Lorsque vous créez un écouteur, vous spécifiez un groupe cible pour son action par défaut. Le trafic est transféré vers le groupe cible spécifié dans la règle de l'écouteur.
- Listener : ressource Elastic Load Balancing ou VPC Lattice qui vérifie les demandes de connexion à l'aide du protocole et du port que vous configurez. Les règles que vous définissez pour un

écouteur déterminent la manière dont Amazon ECS achemine les requêtes vers ses cibles enregistrées.

- Règle : ressource Elastic Load Balancing ou VPC Lattice associée à un écouteur. Une règle définit la manière dont les requêtes sont acheminées et comprend une action, une condition et une priorité.

Considérations

Tenez compte des éléments suivants lors du choix d'un type de déploiement :

- Utilisation des ressources : les déploiements Canary exécutent simultanément des ensembles de tâches d'origine et des ensembles de tâches Canary pendant la période d'évaluation, ce qui augmente l'utilisation des ressources.
- Volume de trafic : assurez-vous que le pourcentage canarien génère un trafic suffisant pour une validation significative de la nouvelle version.
- Complexité de la surveillance : les déploiements de Canary nécessitent une surveillance et une comparaison simultanées des métriques entre deux versions différentes.
- Vitesse de restauration : les déploiements de Canary permettent une restauration rapide en redirigeant le trafic vers l'ensemble de tâches d'origine.
- Atténuation des risques : les déploiements de Canary offrent une excellente atténuation des risques en limitant l'exposition à un faible pourcentage d'utilisateurs.
- Durée du déploiement : les déploiements de Canary incluent des périodes d'évaluation qui prolongent la durée globale du déploiement tout en offrant des opportunités de validation.

Comment fonctionnent les déploiements de Canary

Le processus de déploiement d'Amazon ECS Canary suit une approche structurée en six phases distinctes qui garantissent des mises à jour d'applications sûres et fiables. Chaque phase a un objectif spécifique en validant et en faisant passer votre application de la version actuelle (bleu) à la nouvelle version (vert).

1. Phase de préparation : création de l'environnement vert à côté de l'environnement bleu existant.
2. Phase de déploiement : déploiement de la nouvelle version du service dans un environnement vert. Amazon ECS lance de nouvelles tâches à l'aide de la révision de service mise à jour tandis que l'environnement bleu continue de desservir le trafic de production.

3. Phase de test : validation de l'environnement vert à l'aide du routage du trafic de test. L'Application Load Balancer dirige les requêtes de test vers l'environnement vert tandis que le trafic de production reste en mode bleu.
4. Phase de transfert du trafic des Canaries : transfert du pourcentage configuré du trafic vers la nouvelle révision du service vert pendant la phase canarienne, puis transfert de 100,0 % du trafic vers la révision du service vert
5. Phase de surveillance : surveillance de l'état de l'application, des métriques de performance et des états d'alarme pendant la durée de l'intégration. Une opération de restauration est lancée lorsque des problèmes sont détectés.
6. Phase d'achèvement : finalisez le déploiement en mettant fin à l'environnement bleu.

La phase de transfert de trafic des Canaries suit les étapes suivantes :

- Initiale : le déploiement commence avec 100 % du trafic acheminé vers la version bleue (actuelle) du service. La (nouvelle) révision de service verte reçoit du trafic de test mais aucun trafic de production au départ.
- Transfert de trafic aux Canaries - Il s'agit d'une stratégie de transfert de trafic en deux étapes.
  - Étape 1 : 10,0 % au vert, 90,0 % au bleu
  - Étape 2 : 100,0 % au vert, 0,0 % au bleu
- Temps de cuisson des canaris - Attend pendant une durée configurable (temps de cuisson des canaris) après le changement de trafic des Canaries afin de permettre le suivi et la validation des performances de la nouvelle révision face à l'augmentation de la charge de trafic.
- Hooks du cycle de vie : des fonctions Lambda facultatives ou des crochets de pause peuvent être configurés à différentes étapes du cycle de vie pendant le déploiement afin d'effectuer une validation, une surveillance ou une logique personnalisée automatisées. Hooks configurés pour `PRODUCTION_TRAFFIC_SHIFT` ou `PRE_PRODUCTION_TRAFFIC_SHIFT` sont invoqués à chaque étape du changement de trafic de production.

Étapes du cycle de vie du déploiement

Le processus de déploiement de Canary passe par différentes étapes du cycle de vie, chacune comportant des responsabilités et des points de contrôle de validation spécifiques. La compréhension de ces étapes vous permet de suivre la progression du déploiement et de résoudre les problèmes de manière efficace.

Chaque étape du cycle de vie peut durer jusqu'à 24 heures et, en outre, chaque étape de changement de trafic dans PRODUCTION_TRAFFIC_SHIFT peut durer jusqu'à 24 heures. Nous recommandons que la valeur reste inférieure à 24 heures. Cela est dû au fait que les processus asynchrones ont besoin de temps pour déclencher les hooks. Le système expire, échoue lors du déploiement, puis lance une restauration après 24 heures d'une étape.

CloudFormation les déploiements sont soumis à des restrictions de délai supplémentaires. Tant que la limite de 24 heures reste en vigueur, CloudFormation impose une limite de 36 heures pour l'ensemble du déploiement. CloudFormation fait échouer le déploiement, puis lance une restauration si le processus ne se termine pas dans les 36 heures.

Pour les crochets de pause, vous pouvez configurer le délai d'attente jusqu'à 20 160 minutes (14 jours). Le délai de déploiement global est de 30 jours.

Étapes du cycle de vie

Étapes du cycle de vie	Description	Support Lifecycle Hook
RECONCILE_SERVICE	Cette étape ne se produit que lorsque vous démarrez un nouveau déploiement de service avec plus d’une révision de service dans un état ACTIF.	Oui
PRE_SCALE_UP	La révision de service verte n’a pas commencé. La révision de service bleue gère 100 % du trafic de production. Il n’y a aucun trafic de test.	Oui
SCALE_UP	Le moment où la révision de service verte augmente verticalement jusqu’à 100 % et lance de nouvelles tâches. La révision de service verte ne génère aucun trafic pour le moment.	Non
POST_SCALE_UP	La révision de service verte a commencé. La révision de service bleue gère 100 % du trafic de production. Il n’y a aucun trafic de test.	Oui
TEST_TRAFFIC_SHIFT	Les révisions de service bleues et vertes sont en cours. La révision de service bleue gère 100 % du trafic de production. La révision de	Oui (Lambda uniquement)

Étapes du cycle de vie	Description	Support Lifecycle Hook
	service verte passe de 0 % à 100 % du trafic de test.	
POST_TEST_TRAFFIC_SHIFT	Le transfert du trafic de test est terminé. La révision de service verte gère 100 % du trafic de test.	Oui
MODIFICATION_DE_TRAITEMENT_PRÉALABLE À LA PRODUCTION	Se produit avant chaque étape de transfert du trafic de production. Pour les îles Canaries, cela se produit avant le changement de trafic des Canaries et avant le quart de trafic restant.	Oui
PRODUCTION_TRAFFIC_SHIFT	Le trafic de production de Canary est acheminé vers la version verte et le hook du cycle de vie est invoqué avec un délai d'expiration de 24 heures. La deuxième étape fait passer le trafic de production restant à la révision verte.	Oui (Lambda uniquement)
POST_PRODUCTION_TRAFFIC_SHIFT	Le transfert du trafic de production est terminé.	Oui
BAKE_TIME	Durée pendant laquelle les révisions de service bleues et vertes sont exécutées simultanément.	Non
CLEAN_UP	La révision de service bleue a été complètement réduite à 0 tâches en cours d'exécution. Au terme de cette étape, la révision de service verte devient la révision de service de production.	Non

Paramètres de configuration

Les déploiements Canary nécessitent les paramètres de configuration suivants :

- Pourcentage canarien : pourcentage du trafic à acheminer vers la nouvelle révision du service pendant la phase canarienne. Cela permet de réaliser des tests avec un sous-ensemble contrôlé du trafic de production.
- Durée de cuisson des Canaries : durée d'attente pendant la phase canarienne avant de transférer le trafic restant vers la nouvelle révision du service. Cela laisse le temps de surveiller et de valider la nouvelle version.

Gestion du trafic

Les déploiements Canary utilisent des groupes cibles d'équilibrage de charge pour gérer la distribution du trafic :

- Groupe cible d'origine : contient les tâches de la version stable actuelle et reçoit la majeure partie du trafic.
- Groupe cible Canary : contient les tâches de la nouvelle version et reçoit un faible pourcentage du trafic destiné aux tests.
- Routage pondéré : l'équilibreur de charge utilise des règles de routage pondérées pour répartir le trafic entre les groupes cibles en fonction du pourcentage canarien configuré.

Surveillance et validation

Les déploiements efficaces de Canary reposent sur une surveillance complète :

- Contrôles de santé : les deux ensembles de tâches doivent réussir des contrôles de santé avant de recevoir du trafic.
- Comparaison des métriques : comparez les indicateurs de performance clés entre la version originale et la version Canary, tels que le temps de réponse, le taux d'erreur et le débit.
- Restauration automatique - Configuration CloudWatch alarmes pour déclencher automatiquement la restauration si la version Canary affiche des performances dégradées.
- Validation manuelle : utilisez la période d'évaluation pour examiner manuellement les journaux, les mesures et les commentaires des utilisateurs avant de poursuivre.

Meilleures pratiques pour les déploiements de Canary

Suivez ces bonnes pratiques pour garantir la réussite des déploiements de services avec Canary.

Choisissez les pourcentages de trafic appropriés

Tenez compte des facteurs suivants lorsque vous sélectionnez les pourcentages de trafic des Canaries :

- Commencez petit : commencez avec 5 à 10 % du trafic afin de minimiser l'impact en cas de problème.
- Tenez compte de la criticité des applications : utilisez des pourcentages plus faibles pour les applications critiques et des pourcentages plus élevés pour les services moins critiques.
- Tenez compte du volume de trafic - Assurez-vous que le pourcentage canarien génère un trafic suffisant pour une validation significative.

Fixez des périodes d'évaluation appropriées

Configurez les périodes d'évaluation en fonction des considérations suivantes :

- Prévoyez suffisamment de temps : définissez des périodes d'évaluation suffisamment longues pour recueillir des données de performance significatives, généralement de 10 à 30 minutes.
- Tenez compte des modèles de trafic : tenez compte des modèles de trafic et des périodes de pointe d'utilisation de votre application.
- Équilibrez vitesse et sécurité - Des périodes d'évaluation plus longues fournissent davantage de données mais ralentissent la vitesse de déploiement.

Mettre en œuvre une surveillance complète

Configurez la surveillance pour suivre les performances de déploiement de Canary :

- Indicateurs clés : surveillez le temps de réponse, le taux d'erreur, le débit et l'utilisation des ressources pour les deux ensembles de tâches.
- Alarm-based rollback - Configurer CloudWatch des alarmes pour déclencher automatiquement l'annulation lorsque les métriques dépassent les seuils.
- Analyse comparative - Configurez des tableaux de bord pour comparer les statistiques entre les versions originale et Canary côte à côte.
- Indicateurs commerciaux : incluez des indicateurs spécifiques à l'entreprise, tels que les taux de conversion ou l'engagement des utilisateurs, aux côtés des indicateurs techniques.

Planifiez des stratégies de restauration

Préparez-vous à d'éventuels scénarios de restauration grâce aux stratégies suivantes :

- Annulation automatique : configurez des déclencheurs d'annulation automatiques en fonction des bilans de santé et des mesures de performance.
- Procédures de restauration manuelle : documentez des procédures claires pour la restauration manuelle lorsque les déclencheurs automatiques ne permettent pas de détecter tous les problèmes.
- Tests d'annulation : testez régulièrement les procédures d'annulation pour vous assurer qu'elles fonctionnent correctement en cas de besoin.

Validez minutieusement avant le déploiement

Assurez-vous d'une validation complète avant de procéder aux déploiements de Canary :

- Pre-deployment testing - Testez minutieusement les modifications apportées aux environnements de test avant le déploiement de Canary.
- Configuration du bilan de santé : assurez-vous que les bilans de santé reflètent avec précision l'état de préparation et les fonctionnalités de l'application.
- Validation des dépendances : vérifiez que les nouvelles versions sont compatibles avec les services en aval et en amont.
- Cohérence des données : assurez-vous que les modifications du schéma de base de données et les migrations de données sont rétrocompatibles.

Coordonner la participation de l'équipe

Assurez une coordination efficace de l'équipe lors des déploiements de Canary :

- Fenêtres de déploiement : planifiez les déploiements de Canary pendant les heures ouvrables, lorsque les équipes sont disponibles pour surveiller et intervenir.
- Canaux de communication - Établissez des canaux de communication clairs pour l'état du déploiement et l'escalade des problèmes.
- Attribution des rôles : définissez les rôles et les responsabilités en matière de surveillance, de prise de décision et d'exécution rétroactive.

Ressources requises pour les déploiements d'Amazon ECS Canary

Pour utiliser un déploiement Canary avec transfert de trafic géré, votre service doit utiliser l'une des fonctionnalités suivantes :

- Elastic Load Balancing
- Connexion au service
- Treillis en PVC

La liste suivante fournit une vue d'ensemble de haut niveau de ce que vous devez configurer pour les déploiements Amazon ECS Canary :

- Votre service utilise un équilibreur de charge d'application, un équilibreur de charge réseau, Service Connect ou VPC Lattice. Configurez les ressources appropriées.
  - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
  - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
  - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
  - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.
- Définissez le contrôleur de déploiement de service sur `ECS`.
- Configurez la stratégie de déploiement sur `canary` dans votre définition de service.
- Configurez éventuellement des paramètres supplémentaires tels que :
  - La durée de l'intégration pour le nouveau déploiement
  - Pourcentage de trafic à acheminer vers la nouvelle révision du service pendant la phase canarienne.
  - Durée d'attente pendant la phase canarienne avant de transférer le trafic restant vers la nouvelle révision du service.
  - CloudWatch alarmes pour la restauration automatique
  - Hooks du cycle de vie du déploiement (fonctions Lambda ou crochets de pause qui s'exécutent à des étapes de déploiement spécifiées)

Bonnes pratiques

Suivez ces bonnes pratiques pour réussir les déploiements Amazon ECS lCanary :

- Assurez-vous que votre application peut gérer les deux révisions de service exécutées simultanément.
- Prévoyez une capacité de cluster suffisante pour gérer les deux révisions de service pendant le déploiement.
- Testez vos procédures de restauration avant de les implémenter en production.
- Configurez les surveillances de l’état appropriées qui reflètent précisément l’état de votre application.
- Définissez une durée de l’intégration permettant de tester suffisamment le déploiement vert.
- Mettre en œuvre CloudWatch des alarmes pour détecter automatiquement les problèmes et déclencher des annulations.
- Utilisez les hooks de cycle de vie pour effectuer des tests automatisés à chaque étape du déploiement.
- Commencez par de faibles pourcentages de canaris (5 à 10 %) afin de minimiser l'impact en cas de problème.
- Fixez des périodes d'évaluation appropriées qui laissent suffisamment de temps pour recueillir des données de performance significatives.
- Mettez en œuvre une surveillance complète avec CloudWatch alarmes pour les déclencheurs d'annulation automatisés.
- Configurez des bilans de santé qui reflètent avec précision l'état de préparation et les fonctionnalités de votre application.
- Surveillez à la fois les indicateurs techniques (temps de réponse, taux d'erreur) et les indicateurs commerciaux lors de l'évaluation.
- Assurez-vous que votre application peut gérer le fractionnement du trafic sans problèmes de session ou d'état.
- Planifiez des procédures de restauration et testez-les régulièrement pour vous assurer qu'elles fonctionnent en cas de besoin.
- Planifiez les déploiements de Canary pendant les heures de bureau, lorsque les équipes peuvent surveiller et réagir.
- Validez minutieusement les modifications dans les environnements de test avant le déploiement de Canary.

- Documentez des procédures claires pour les interventions manuelles et les décisions d'annulation.

Création d'un déploiement Amazon ECS Canary

En utilisant les déploiements Amazon ECS Canary, vous pouvez transférer un faible pourcentage du trafic vers votre nouvelle révision de service (la « version Canary »). Validez le déploiement, puis déplacez le trafic restant en une seule fois après un intervalle spécifié. Cette approche vous permet de tester de nouvelles fonctionnalités avec un minimum de risques avant le déploiement complet.

Conditions préalables

Effectuez les opérations suivantes avant de démarrer un déploiement Canary.

1. Configurez les autorisations appropriées.
   - Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.
   - Si vous utilisez VPC Lattice, pour plus d'informations sur les autorisations VPC Lattice, consultez `AmazonECSInfrastructureRolePolicyForVpcLattice`.
   - Pour plus d'informations sur les autorisations Lambda, consultez la section Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.
2. Les déploiements d'Amazon ECS Canary nécessitent que votre service utilise l'une des fonctionnalités suivantes : Configurez les ressources appropriées.
   - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
   - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
   - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
   - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

Procédure

Vous pouvez utiliser la console ou AWS CLI pour créer un service de déploiement Amazon ECS Canary.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource à partir de laquelle vous lancez le service.

Pour démarrer un service à partir de	Étapes
Clusters	a. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service. La page des détails du cluster s'ouvre. b. Dans l'onglet Services choisissez Create (Créer).
Définition de tâche	a. Sur la page Définitions de tâche, sélectionnez la définition de tâche. b. Dans le menu Déployer, choisissez Créer un service.

   La page Créer un service s'affiche.
3. Sous Détails du service, procédez comme suit :

   a. Pour Famille de définition de tâche, choisissez la définition de tâche à utiliser. Ensuite, pour Révision de définition de tâche, saisissez la révision à utiliser.

   b. Pour Service name (Nom du service), saisissez un nom pour votre service.
4. Pour exécuter le service dans un cluster existant, pour Cluster existant, sélectionnez le cluster. Pour exécuter le service dans un nouveau cluster, choisissez Créer un cluster
5. Choisissez comment vos tâches sont distribuées dans votre infrastructure de cluster. Sous Configuration de calcul, choisissez votre option.

Options de calcul	Étapes	
Stratégie de fournisseur de capacité	a. Sous Options de calcul, choisissez Stratégie de fournisseur de capacité. b. Choisissez une stratégie : • Pour utiliser une stratégie de fournisseur de capacité, choisissez Use cluster default (Utiliser la stratégie par défaut du cluster). • Si votre cluster ne possède pas de stratégie de fournisseur de capacité par défaut ou si vous souhaitez utiliser une stratégie personnalisée, sélectionnez Utiliser une stratégie personnalisée ou Ajouter une stratégie de fournisseur de capacité et définissez votre stratégie de fournisseur de capacité personnalisée en spécifiant une Base, un Fournisseur de capacité et un Poids.	

Options de calcul	Étapes
	Note Pour utiliser un fournisseur de capacité dans une stratégie, le fournisseur de capacité doit être associé au cluster.
Type de lancement	a. Dans Compute options (Options de calcul), sélectionnez Launch type (Type de lancement). b. Pour Launch type (Type de lancement), choisissez un type de lancement. c. (Facultatif) Lorsque Fargate est spécifié, pour Version de plateforme, spécifiez la version de plateforme à utiliser. Si vous ne spécifiez aucune version de plateforme, la version `LATEST` est utilisée.

6. Sous Configuration du déploiement, procédez comme suit :

   a. Pour Type de service, choisissez Replica.

   b. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.

c. Pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage des services entre zones de disponibilité, sélectionnez Rééquilibrage des services entre zones de disponibilité.

d. Pour Période de grâce pour la surveillance de l'état, saisissez la durée (en secondes) pendant laquelle le planificateur de service ignore les surveillances de l'état de Elastic Load Balancing, VPC Lattice et des conteneurs qui ne sont pas saines après le démarrage initial d'une tâche. Si vous ne spécifiez aucune valeur pour la période de grâce de surveillance de l'état, la valeur par défaut 0 est utilisée.

7. En dessous Configuration du déploiement, configurez les paramètres de déploiement de Canary :

   a. Pour Stratégie de déploiement, choisissez Canari.

   b. Pour Pourcentage canarien, entrez le pourcentage du trafic à passer à la révision du service vert lors de la première étape (par exemple, 10 % pour le trafic initial des Canaries).

   c. Pour L'heure de la cuisson aux Canaries, entrez le temps d'attente en minutes avant de transférer le trafic restant vers la révision du service vert.

   d. Pour Temps de cuisson, entrez le nombre de minutes pendant lesquelles les révisions de service bleues et vertes seront exécutées simultanément après le dernier changement de trafic avant la fin de la révision bleue.

   e. (Facultatif) Configurez les hooks du cycle de vie pour qu'ils s'exécutent à des étapes spécifiques du déploiement. En dessous Crochets liés au cycle de vie, choisissez Ajouter.

      Pour les crochets Lambda :

      - Pour Type de cible, choisissez Lambda.
      - Pour Fonction Lambda, saisissez le nom ou l'ARN de la fonction.
      - Pour Rôle, sélectionnez le rôle IAM qui dispose de l'autorisation requise pour invoquer la fonction Lambda.
      - Pour Étapes du cycle de vie, sélectionnez les étapes dans lesquelles la fonction Lambda doit être exécutée.

      Pour les crochets de pause :

- Pour Type de cible, choisissez Pause.
- Pour étapes du cycle de vie, sélectionnez les étapes pendant lesquelles le déploiement doit être suspendu.
- (Facultatif) Pour Délai d'attente, configurez la durée et l'action du délai d'attente.

8. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

   a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

      Pour que le logiciel restaure automatiquement le dernier état de déploiement terminé lorsque le disjoncteur de déploiement définit le déploiement comme ayant échoué, sélectionnez Restauration en cas d’échec.

   b. Pour arrêter un déploiement en fonction des statistiques de l'application, sélectionnez Utiliser CloudWatch alarme (s). Puis, à partir de CloudWatch nom de l'alarme, choisissez les alarmes. Pour créer une nouvelle alarme, accédez au CloudWatch console.

      Pour que le logiciel ramène automatiquement le déploiement au dernier état de déploiement terminé lorsqu'un CloudWatch l'alarme met le déploiement en état d'échec, sélectionnez Annulation en cas de panne.

9. (Facultatif) Pour interconnecter votre service à l’aide de Service Connect, développez Service Connect, puis spécifiez les éléments suivants :

   a. Sélectionnez Activer Service Connect.

   b. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

      - Si votre service exécute une application client réseau qui doit uniquement se connecter à d’autres services de l’espace de noms, sélectionnez Côté client uniquement.
      - Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

c. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service. Il peut s'agir d'un espace de noms créé séparément dans le même Région AWS dans votre Compte AWS ou un espace de noms dans la même région qui est partagé avec votre compte à l'aide de AWS Resource Access Manager (AWS RAM). Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

d. (Facultatif) Configurez les règles d'en-tête de trafic de test pour les déploiements Canary. Sous Routage du trafic de test, spécifiez les éléments suivants :

   i. Sélectionnez Activer les règles d’en-tête du trafic de test pour acheminer des requêtes spécifiques vers la révision de service verte pendant les tests.

   ii. Pour Règles de correspondance des en-têtes, configurez les critères de routage du trafic de test suivants :

   - Nom de l’en-tête : saisissez le nom de l’en-tête HTTP correspondant (par exemple, `X-Test-Version` ou `User-Agent`).
   - Type de correspondance : choisissez les critères de correspondance :
     - Correspondance exacte : acheminer les requêtes dont la valeur d’en-tête correspond exactement à la valeur spécifiée
     - En-tête présent : acheminer les requêtes contenant l’en-tête spécifié, quelle que soit sa valeur
     - Correspondance de modèle : acheminer les requêtes dont la valeur d’en-tête correspond à un modèle spécifié
   - Valeur d’en-tête (si vous utilisez une correspondance exacte ou une correspondance de modèle) : saisissez la valeur ou le modèle à comparer.

   Vous pouvez ajouter plusieurs règles de correspondance d’en-têtes pour créer une logique de routage complexe. Les requêtes correspondant à l’une des règles configurées seront acheminées vers la version verte du service à des fins de test.

   iii. Choisissez Ajouter une règle d’en-tête pour configurer des conditions de correspondance d’en-tête supplémentaires.

> **Note**
>
> Les règles d'en-tête du trafic de test vous permettent de valider les nouvelles fonctionnalités avec un trafic contrôlé avant de procéder au déploiement complet. Cela vous permet de tester la révision de service verte avec des requêtes spécifiques (telles que celles provenant d'outils de test internes ou d'utilisateurs bêta) tout en maintenant un flux de trafic normal vers la révision de service bleue.

e. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux des conteneurs à CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Amazon Data Firehose : configurez la tâche pour envoyer des journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Elles envoient les journaux vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Amazon Kinesis Data Streams : configurez la tâche pour envoyer des journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Elles permettent d'envoyer les journaux vers un flux Kinesis Data Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.
   - Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
   - Simple Storage Service (Amazon S3) : configurez la tâche pour envoyer des journaux de conteneur à un compartiment Simple Storage Service (Amazon S3). Les options du

pilote de journal par défaut sont définies par défaut, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

10. (Facultatif) Configurer équilibrage de charge pour le déploiement de Canary.

Types Elastic Load Balancing	Étapes	
Application Load Balancer	a. Pour Type d'équilibreur de charge, choisisse z Application Load Balancer. b. Choisissez Create a new load balancer (Créer un équilibreur de charge) pour créer un équilibre ur de charge Application Load Balancer ou Use an existing load balancer (Utiliser un équilibreur de charge existant) pour sélectionner un équilibre ur de charge Application Load Balancer existant. c. Pour Conteneur, choisissez le conteneur qui héberge le service. d. Pour Load balancer name (nom d'équilibreur de charge), saisissez un nom unique. e. Pour Listener (Écouteur), saisissez un port et un protocole sur lesquels Application Load Balancer doit écouter les	

Types Elastic Load Balancing	Étapes	
	demandes de connexion . Par défaut, l'équilibreur de charge est configuré pour utiliser le port 80 et HTTP. • Pour Règle de production, saisissez l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service de production (bleu). • Pour Règle de test, saisissez l'ordre d'évaluation et le modèle de chemin de la règle. Cette règle concerne le trafic de révision de votre service test (vert). f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Applicati	

Types Elastic Load Balancing	Étapes	
	on Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Applicati on Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscri ption, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`. La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Applicati on Load Balancer doit envoyer périodiqu ement des requêtes pour vérifier l'état de la connexion	

Types Elastic Load Balancing	Étapes	
	entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Pour Autre nom de groupe, entrez le nom du groupe cible pour la révision de votre service de test.	

Types Elastic Load Balancing	Étapes	
Network Load Balancer	a. Pour Load balancer type (Type d'équilibreur de charge), sélectionnez Network Load Balancer. b. Dans le champ Load Balancer (Équilibreur de charge), choisissez Attacher à un Network Load Balancer existant. c. Pour Choose container to load balance (Choisissez le conteneur pour équilibrer la charge), choisissez le conteneur qui héberge le service. d. Pour Écouteur de production, choisissez le port d’écouteur de production et le protocole d’écouteur de production. Il s'agit de l'écouteur pour le trafic de révision de votre service de production. e. Pour Port de l’écouteur de test, choisissez le port de l’écouteur de test et le protocole de l’écouteur de test.	

Types Elastic Load Balancing	Étapes	
	Il s'agit de l'écouteur pour le trafic de révision de votre service de test. f. Pour Groupe cible, configurez les éléments suivants : • Pour Target group name (Nom du groupe cible), saisissez un nom et un protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. • Pour Protocole, choisissez le protocole pour le groupe cible vers lequel Network Load Balancer achemine les requêtes. Par défaut, le groupe cible achemine les requêtes vers le premier conteneur défini dans votre définition de tâche. • Pour Délai de désinscription, entrez le nombre de secondes pendant lequel l'équilibreur de charge doit changer l'état cible en `UNUSED`.	

Types Elastic Load Balancing	Étapes	
	La durée par défaut est 300 secondes. • Pour Health check path (Chemin de surveillance de l'état), saisissez un chemin qui existe dans votre conteneur où Application Load Balancer doit envoyer périodiquement des requêtes pour vérifier l'état de la connexion entre Application Load Balancer et le conteneur. Le répertoire par défaut est le répertoire racine (/). • Dans Nom de groupe alternatif, saisissez le nom du groupe cible pour votre révision de service de test (vert).	

11. (Facultatif) Pour utiliser VPC Lattice pour gérer le transfert de trafic lors des déploiements de Canary, développez Réseau en PVC - facultatif et sélectionnez Utiliser VPC Lattice, puis procédez comme suit :

    a. Pour VPC, choisissez le VPC pour vos ressources VPC Lattice. Il doit s'agir du même VPC que votre service Fargate.

    b. Pour Rôle de l'infrastructure, choisissez le rôle IAM de l'infrastructure Amazon ECS qu'Amazon ECS utilise pour gérer les ressources de votre VPC Lattice, ou choisissez Création d'un nouveau rôle d'infrastructure.

c. Pour Nom du port du conteneur, choisissez le nom du port dans la définition de tâche qui correspond au port de conteneur qui reçoit le trafic de VPC Lattice.

d. Pour Service VPC Lattice, choisissez Créer un nouveau service ou Utiliser un service existant. Pour Nom du service, entrez un nom unique pour le service.

e. Pour auditeur, choisissez Créer un nouvel auditeur ou Utiliser un écouteur existant. Pour Port, entrez un port compris entre 1 et 65535. Pour Protocole, choisissez le protocole (par exemple, HTTP).

f. En dessous Règle relative à l'auditeur de production, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire (maximum 200 caractères). Amazon ECS utilise cette règle pour acheminer le trafic de production.

g. (Facultatif, recommandé) Sous Règle de test pour les auditeurs, entrez un Nom, une Priorité de 1 à 100, et un Schéma de trajectoire. Amazon ECS utilise cette règle pour acheminer le trafic de test vers la révision du service vert.

h. Pour Groupes cibles, choisissez Création de deux nouveaux groupes cibles, Créez un groupe cible vert, ou Utiliser deux groupes cibles existants. Pour un nouveau groupe cible, entrez Nom du groupe cible, choisissez un Protocole, entrez un Port de 1 à 65535, choisissez Protocole de bilan de santé, et entrez un Parcours de contrôle de santé. Pour Nom du groupe cible en vert, entrez un nom ; le groupe cible vert utilise la même configuration que le groupe cible bleu.

12. (Facultatif) Pour vous aider à identifier votre service et vos tâches, développez Tags (balises), puis configurez vos balises.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises de définition des tâches, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Définitions de tâches.

    Pour qu'Amazon ECS balise automatiquement toutes les tâches nouvellement lancées avec le nom du cluster et les balises d'un service, sélectionnez Activer les balises gérées Amazon ECS, puis pour Propager des balises à partir de, choisissez Service.

    Ajoutez ou supprimez une balise.

    - [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.

- [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.

13. Cliquez sur Créer.

AWS CLI

1. Créez un fichier nommé `canary-service-definition.json` avec le contenu suivant.

   Remplacez le *`user-input`* avec tes valeurs.

```
{
  "serviceName": "myCanaryService",
  "cluster": "arn:aws:ecs:us-west-2:123456789012:cluster/sample-fargate-
cluster",
  "taskDefinition": "sample-fargate:1",
  "desiredCount": 5,
  "launchType": "FARGATE",
  "networkConfiguration": {
    "awsvpcConfiguration": {
      "subnets": [
        "subnet-09ce6e74c116a2299",
        "subnet-00bb3bd7a73526788",
        "subnet-0048a611aaec65477"
      ],
      "securityGroups": [
        "sg-09d45005497daa123"
      ],
      "assignPublicIp": "ENABLED"
    }
  },
  "deploymentController": {
    "type": "ECS"
  },
  "deploymentConfiguration": {
    "strategy": "CANARY",
    "maximumPercent": 200,
    "minimumHealthyPercent": 100,
    "canaryConfiguration" : {
        "canaryPercent" : 5.0,
        "canaryBakeTime" : 10
    },
    "bakeTimeInMinutes": 10,
    "alarms": {
```

```
      "alarmNames": [
        "myAlarm"
      ],
      "rollback": true,
      "enable": true
    }
  },
  "loadBalancers": [
    {
      "targetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/blue-target-group/54402ff563af1197",
      "containerName": "fargate-app",
      "containerPort": 80,
      "advancedConfiguration": {
        "alternateTargetGroupArn": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:targetgroup/green-target-group/cad10a56f5843199",
        "productionListenerRule": "arn:aws:elasticloadbalancing:us-
west-2:123456789012:listener-rule/app/my-canary-
demo/32e0e4f946c3c05b/9cfa8c482e204f7d/831dbaf72edb911",
        "roleArn": "arn:aws:iam::123456789012:role/LoadBalancerManagementforECS"
      }
    }
  ]
}
```

2. Exécutez `create-service`.

```
aws ecs create-service --cli-input-json file://canary-service-definition.json
```

Étapes suivantes

Après avoir configuré votre déploiement Canary, procédez comme suit :

- Mettez à jour le service pour démarrer le déploiement. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il suit le modèle Canary :
  - La révision de service verte est créée et augmentée horizontalement
  - Un faible pourcentage du trafic (canaris) est transféré vers la version verte
  - Le système attend l'intervalle canarien spécifié
  - Le trafic restant est transféré en une seule fois vers la version verte

- Au terme de la durée de l’intégration, la révision bleue est résiliée

Terminologie du déploiement

Les termes suivants sont utilisés dans la documentation de déploiement d'Amazon ECS :

Blue-green déploiement

Stratégie de déploiement qui crée un nouvel environnement (vert) parallèlement à l'environnement existant (bleu), puis fait passer le trafic du bleu au vert après validation.

Déploiement Canary

Stratégie de déploiement qui achemine un faible pourcentage du trafic vers une nouvelle version tout en conservant la majorité sur la version stable à des fins de validation.

Déploiement linéaire

Stratégie de déploiement qui déplace progressivement le trafic de l'ancienne version vers la nouvelle version par incréments égaux au fil du temps.

Déploiement progressif

Stratégie de déploiement qui remplace les instances de l'ancienne version par des instances de la nouvelle version, une par une.

Ensemble de tâches

Ensemble de tâches qui exécutent la même définition de tâches au sein d'un service lors d'un déploiement.

Groupe cible

Regroupement logique de cibles recevant le trafic d'un équilibreur de charge pendant les déploiements.

Contrôleur de déploiement

La méthode utilisée pour déployer de nouvelles versions de votre service, comme Amazon ECS, CodeDeploy, ou des contrôleurs externes.

Restauration

Processus de retour à une version précédente de votre application lorsque des problèmes sont détectés lors du déploiement.

Déploiement de services Amazon ECS à l’aide d’un contrôleur tiers

Le type de déploiement externe vous permet d'utiliser n'importe quel contrôleur de déploiement tiers pour un contrôle complet du processus de déploiement pour un service Amazon ECS service. Les détails pour votre service sont gérés par les actions d'API de gestion de service (`CreateService`, `UpdateService` et `DeleteService`) ou par les actions d'API de gestion des tâches définies (`CreateTaskSet`, `UpdateTaskSet`, `UpdateServicePrimaryTaskSet` et `DeleteTaskSet`). Chaque action d'API gère un sous-ensemble de paramètres de définition de service.

L'action d'API `UpdateService` met à jour le nombre souhaité et les paramètres de période de délai supplémentaire des surveillances de l'état pour un service. Si l’option de calcul, la version de plateforme, les détails de l’équilibreur de charge, la configuration réseau ou la définition de tâche doivent être mis à jour, vous devez créer un jeu de tâches.

L'action d'API `UpdateTaskSet` met à jour uniquement le paramètre d'échelle pour un ensemble de tâches.

L'action d'API `UpdateServicePrimaryTaskSet` modifie l'ensemble de tâches dans un service qui est l'ensemble de tâches principal. Lorsque vous appelez l'action d'API `DescribeServices`, elle renvoie tous les champs spécifiés pour un ensemble de tâches principal. Si l'ensemble de tâches principal d'un service est mis à jour, toutes les valeurs de paramètres de l'ensemble de tâches qui existe sur l'ensemble de tâches principal différent de l'ancienne tâche principale dans un service sont mises à jour vers la nouvelle valeur lorsqu'un ensemble de tâches principal est défini. Si aucun ensemble de tâches principale n'est défini pour un service, les champs de l'ensemble de tâches sont vides pour la description de service.

Considérations relatives au déploiement externe

Tenez compte des éléments suivants lorsque vous utilisez le type de déploiement externe :

- Les types d'équilibreur de charge pris en charge sont un Application Load Balancer ou un Network Load Balancer.
- Les types de contrôleurs de déploiement Fargate ou `EXTERNAL` ne prennent pas en charge la stratégie de planification `DAEMON`.
- Demande AutoScaling l'intégration à Amazon ECS prend uniquement en charge le service Amazon ECS en tant que cible. La dimension de capacité de mise à l’échelle prise en charge pour Amazon ECS est `ecs:service:DesiredCount`, soit le nombre de tâches d’un service Amazon ECS. Il n'y a pas d'intégration directe entre l'application AutoScaling et des ensembles de tâches Amazon

ECS. Les jeux de tâches Amazon ECS calculent le `ComputedDesiredCount` en fonction du `DesiredCount` du service Amazon ECS.

Workflow de déploiement externe

Voici le flux de travail de base pour la gestion d'un déploiement externe sur Amazon ECS.

Pour gérer un service Amazon ECS service à l'aide d'un contrôleur de déploiement externe

1. Créez un service Amazon ECS service. Le seul paramètre requis est le nom de service. Vous pouvez spécifier les paramètres suivants lors de la création d'un service à l'aide d'un contrôleur de déploiement externe. Tous les autres paramètres de service sont spécifiés lors de la création d'un ensemble de tâches dans le service.

   `serviceName`

   Type : chaîne

   Obligatoire : oui

   Nom de votre service. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés. Dans un cluster, les noms de service doivent être uniques, mais certains services peuvent avoir des noms similaires dans des clusters différents d'une même région ou de plusieurs régions.

   `desiredCount`

   Le nombre d'instances sur la définition de tâche d'ensemble de tâches spécifié à placer et à continuer d'exécuter au sein du service.

   `deploymentConfiguration`

   Paramètres de déploiement facultatifs qui contrôlent le nombre de tâches exécutées durant le déploiement et la commande d'arrêt et de démarrage des tâches.

   `tags`

   Type : tableau d'objets

   Obligatoire : non

   Métadonnées que vous appliquez au service pour faciliter le classement et l'organisation. Chaque étiquette est constituée d'une clé et d'une valeur facultative que vous définissez.

Lorsqu'un service est supprimé, les étiquettes sont également supprimées. Un maximum de 50 balises peut être appliqué au service. Pour plus d'informations, consultez Balisage des ressources Amazon ECS.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`enableECSManagedTags`

Spécifie si utiliser les identifications gérées par Amazon ECS pour les tâches dans le service. Pour plus d'informations, consultez Mise à jour des balises pour la facturation.

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE`

Obligatoire : non

Indique s'il faut copier les étiquettes de la définition de tâche ou du service dans les tâches du service. Si aucune valeur n'est spécifiée, les étiquettes ne sont pas copiées. Les étiquettes

ne peuvent être copiées dans les tâches du service que lors de la création du service. Pour ajouter des balises à une tâche après la création du service ou de la tâche, utilisez l'action d'API `TagResource`.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`schedulingStrategy`

Stratégie de planification à utiliser. Les services utilisant un contrôleur de déploiement externe prennent uniquement en charge la stratégie de planification `REPLICA`.

`placementConstraints`

Tableau d'objets de contraintes de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier un maximum de 10 contraintes par tâche (cette limite comprend les contraintes de la définition de tâche et celles spécifiées lors de l'exécution). Si vous utilisez Fargate, les contraintes de placement des tâches ne sont pas prises en charge.

`placementStrategy`

Objets de stratégie de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier jusqu'à quatre règles de stratégie par service.

Voici un exemple de définition de service pour la création d'un service qui utilise un contrôleur de déploiement externe.

```
{
    "cluster": "",
    "serviceName": "",
    "desiredCount": 0,
    "role": "",
    "deploymentConfiguration": {
        "maximumPercent": 0,
        "minimumHealthyPercent": 0
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
```

```
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION"
}
```

2. Créez un ensemble de tâches initial. L'ensemble de tâches contient les détails suivants pour votre service :

   `taskDefinition`

   La définition de tâche pour les tâches dans l'ensemble de tâches à utiliser.

   `launchType`

   Type : String

   Valeurs valides : `EC2` | `FARGATE` | `EXTERNAL`

   Obligatoire : non

   Type de lancement sur lequel vous pouvez exécuter votre service. Si aucun type de lancement n'est spécifié, the `capacityProviderStrategy` par défaut est utilisé par défaut.

   Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

   Si un `launchType` est spécifié, le paramètre `capacityProviderStrategy` doit être omis.

`platformVersion`

Type : chaîne

Obligatoire : non

Version de la plateforme sur laquelle s'exécutent vos tâches dans le service. Une version de plateforme est spécifiée uniquement pour les tâches utilisant le type de lancement Fargate. Si vous ne spécifiez aucune valeur, la dernière version (`LATEST`) est utilisée par défaut.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Lorsque vous spécifiez `LATEST` comme version de plateforme lors de l'exécution d'une tâche ou de la création d'un service, vous obtenez la version de plateforme la plus récente pour vos tâches. Lorsque vous mettez à l'échelle votre service, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel du service. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.

> **Note**
>
> Les versions de plateforme ne sont pas spécifiées pour les tâches utilisant le type de lancement EC2.

`loadBalancers`

Un objet d'équilibreur de charge qui représente l'équilibreur de charge à utiliser avec votre service. Lorsque vous utilisez un contrôleur de déploiement externe, seuls les équilibreurs de charge Application Load Balancer et les Network Load Balancer sont pris en charge. Si vous utilisez un équilibreur de charge Application Load Balancer, seul un groupe cible d'équilibreur de charge Application Load Balancer est autorisé par tâche.

L'extrait de code suivant montre un exemple d'objet `loadBalancer` à utiliser.

```
"loadBalancers": [
        {
```

```
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
]
```

> **Note**
>
> Lorsque vous spécifiez un objet `loadBalancer`, vous devez spécifier `targetGroupArn` et omettre les paramètres `loadBalancerName`.

`networkConfiguration`

Configuration réseau du service. Ce paramètre est obligatoire pour les définitions de tâches qui utilisent le mode réseau `awsvpc` afin de recevoir leur propre interface réseau Elastic. Il n'est pas pris en charge avec les autres modes réseau. Pour plus d'informations sur la mise en réseau pour Fargate, consultez la section Options de mise en réseau des tâches Amazon ECS pour Fargate.

`serviceRegistries`

Les détails des enregistrements de découverte de service à attribuer à ce service. Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.

`scale`

Pourcentage à virgule flottante du nombre souhaité de tâches à placer et à continuer d'exécuter dans l'ensemble de tâches. La valeur est spécifiée en tant que pourcentage total de `desiredCount` d'un service. Les valeurs acceptées sont des nombres compris entre 0 et 100.

Voici un exemple JSON pour créer un ensemble de tâches pour un contrôleur de déploiement externe.

```
{
    "service": "",
    "cluster": "",
    "externalId": "",
```

```
    "taskDefinition": "",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "scale": {
        "value": null,
        "unit": "PERCENT"
    },
    "clientToken": ""
}
```

3. Lorsque des modifications de service sont nécessaires, utilisez l'action d'API `CreateTaskSet`, `UpdateService` ou `UpdateTaskSet`, en fonction des paramètres que vous mettez à jour. Si vous avez créé un ensemble de tâches, utilisez le paramètre `scale` pour chaque ensemble de tâches dans un service afin de déterminer le nombre de tâches à continuer d'exécuter dans le service. Par exemple, si vous disposez d'un service qui contient `tasksetA` et que vous créez un `tasksetB`, vous pouvez tester la validité de `tasksetB` avant de vouloir lui transférer du trafic de production. Vous pouvez régler l'`scale` pour les deux ensembles de tâches sur `100`, et une fois prêt à transférer tout le trafic de production sur `tasksetB`, vous pouvez mettre à jour l'`scale` de `tasksetA` sur `0` pour le diminuer.

CodeDeploy blue/green déploiements pour Amazon ECS

Nous vous recommandons d'utiliser Amazon ECS blue/green déploiement. Pour plus d’informations, consultez Création d'un Amazon ECS blue/green déploiement.

Le blue/green le type de déploiement utilise le blue/green modèle de déploiement contrôlé par CodeDeploy. Ce type de déploiement permet de vérifier un nouveau déploiement d'un service avant d'y envoyer du trafic de production. Pour plus d'informations, voir Qu'est-ce que CodeDeploy dans le AWS CodeDeploy Guide de l'utilisateur.Validez l'état d'un service Amazon ECS avant le déploiement

Le trafic peut changer de trois manières au cours d'une blue/green déploiement :

- Canary : le trafic est transféré en deux incréments. Vous pouvez choisir parmi les options canary prédéfinies qui définissent le pourcentage de trafic déplacé vers l'ensemble de tâches mises à jour dans le premier incrément, et l'intervalle (en minutes) avant que le trafic restant soit déplacé dans le second incrément.
- Linéaire : le trafic est transféré en incréments égaux, avec un nombre de minutes égal entre chaque incrément. Vous pouvez choisir parmi les options linéaires prédéfinies qui définissent le pourcentage de trafic déplacé pour chaque incrément et le nombre de minutes entre chaque incrément.
- All-at-once — Tout le trafic est transféré de l'ensemble de tâches d'origine à l'ensemble de tâches mis à jour en une seule fois.

Les éléments suivants sont des composants de CodeDeploy qu'Amazon ECS utilise lorsqu'un service utilise le blue/green type de déploiement :

CodeDeploy application

Une collection de CodeDeploy ressources. Comprend un ou plusieurs groupes de déploiement.

CodeDeploy groupe de déploiement

Paramètres de déploiement. Regroupe les éléments suivants :

- Cluster et service Amazon ECS
- Groupe cible de l'équilibreur de charge et informations sur l'écouteur
- Stratégie de restauration du déploiement
- Paramètres de réacheminement du trafic
- Paramètres d'arrêt de la révision initiale
- Configuration de déploiement
- CloudWatch configuration des alarmes qui peut être configurée pour arrêter les déploiements
- SNS ou CloudWatch Paramètres des événements pour les notifications

Pour de plus amples informations, veuillez consulter Utilisation des groupes de déploiement dans CodeDeploy dans le Guide de l'utilisateur AWS CodeDeploy .

CodeDeploy configuration de déploiement

Spécifie comment CodeDeploy achemine le trafic de production vers votre ensemble de tâches de remplacement lors d'un déploiement. Les configurations de déploiement linéaire et canary prédéfinies suivantes sont disponibles. Vous pouvez également créer des déploiements canary et linéaires personnalisés. Pour obtenir de plus amples informations, consultez Utilisation des configurations de déploiement dans le Guide de l'utilisateur AWS CodeDeploy .

- CodeDeployDefault.ECSAllAtOnce: déplace tout le trafic vers le conteneur Amazon ECS mis à jour en une seule fois
- CodeDeployDefault.ECSLinear10PercentEvery1Minutes: déplace 10 % du trafic toutes les minutes jusqu'à ce que tout le trafic soit modifié.
- CodeDeployDefault.ECSLinear10PercentEvery3Minutes: déplace 10 % du trafic toutes les 3 minutes jusqu'à ce que tout le trafic soit modifié.
- CodeDeployDefault.ECSCanary10Percent5Minutes: déplace 10 % du trafic lors du premier incrément. Les 90 % restants sont déployés cinq minutes plus tard.
- CodeDeployDefault.ECSCanary10Percent15Minutes: déplace 10 % du trafic lors du premier incrément. Les 90 % restants sont déployés 15 minutes plus tard.

Révision

Une révision est CodeDeploy fichier de spécifications de l'application (AppSpec fichier). Dans le AppSpec fichier, vous spécifiez l'ARN complet de la définition de tâche ainsi que le conteneur et le port de votre ensemble de tâches de remplacement où le trafic doit être acheminé lors de la création d'un nouveau déploiement. Le nom du conteneur doit être un des noms de conteneur référencés dans votre définition de tâche. Si la configuration réseau ou la version de la plate-forme ont été mises à jour dans la définition du service, vous devez également spécifier ces détails dans le AppSpec dossier. Vous pouvez également spécifier les fonctions Lambda à exécuter au cours des événements de cycle de vie du déploiement. Les fonctions Lambda vous permettent d'exécuter des tests et de renvoyer des métriques pendant le déploiement. Pour plus d'informations, voir AppSpec Référence du dossier dans le AWS CodeDeploy Guide de l'utilisateur.

Considérations

Tenez compte des points suivants lorsque vous utilisez le blue/green type de déploiement :

- Lorsqu'un service Amazon ECS utilisant le blue/green Le type de déploiement est initialement créé, un ensemble de tâches Amazon ECS est créé.
- Vous devez configurer le service pour utiliser un équilibreur de charge Application Load Balancer ou un Network Load Balancer. Les exigences de l'équilibreur de charge sont les suivantes :
  - Vous devez ajouter un port d'écoute de production à l'équilibreur de charge, utilisé pour acheminer le trafic de production.
  - Vous pouvez également ajouter à l'équilibreur de charge un port d'écoute de test, utilisé pour acheminer le trafic de test. Si vous spécifiez un écouteur de test, CodeDeploy achemine votre trafic de test vers la tâche de remplacement définie lors d'un déploiement.
  - Les ports d'écoute de production et de test doivent appartenir au même équilibreur de charge.
  - Vous devez définir un groupe cible pour l'équilibreur de charge. Le groupe cible achemine le trafic vers la tâche initiale dans un service via le port d'écoute de production.
  - Lorsqu'un Network Load Balancer est utilisé, seule la configuration de déploiement `CodeDeployDefault.ECSAllAtOnce` est prise en charge.
- Pour les services configurés pour utiliser la mise à l'échelle automatique des services et blue/green type de déploiement, la mise à l'échelle automatique n'est pas bloquée pendant un déploiement, mais le déploiement peut échouer dans certaines circonstances. Ce comportement est décrit plus en détail ci-dessous.

  - Si un service évolue et qu'un déploiement démarre, l'ensemble de tâches vert est créé et CodeDeploy attendra jusqu'à une heure pour que la tâche verte atteigne un état stable et ne modifiera pas le trafic tant que ce n'est pas le cas.
  - Si un service est en cours de traitement blue/green lors du déploiement et d'un événement de dimensionnement, le trafic continuera de changer pendant 5 minutes. Si le service n'atteint pas un état stable dans les 5 minutes, CodeDeploy arrêtera le déploiement et le marquera comme ayant échoué.
- Les tâches qui utilisent Fargate ou les types de contrôleur de déploiement `CODE_DEPLOY` ne prennent pas en charge la stratégie de planification `DAEMON`.
- Lorsque vous créez pour la première fois un CodeDeploy application et groupe de déploiement, vous devez spécifier les éléments suivants :
  - Vous devez définir deux groupes cibles pour l'équilibreur de charge. L'un d'eux doit être le groupe cible initial défini pour l'équilibreur de charge lors de la création du service Amazon ECS service. La seule exigence relative au second groupe cible est qu'il ne peut pas être associé à un autre équilibreur de charge que celui utilisé par le service.
- Lorsque vous créez un CodeDeploy déploiement pour un service Amazon ECS, CodeDeploy crée un ensemble de tâches de remplacement (ou ensemble de tâches vert) lors du déploiement. Si vous avez ajouté un écouteur de test à l'équilibreur de charge, CodeDeploy achemine votre trafic de test vers l'ensemble de tâches de remplacement. Cela indique le moment où vous pouvez exécuter les tests de validation. Ensuite, CodeDeploy réachemine le trafic de production depuis l'ensemble de tâches initial vers l'ensemble des tâches de remplacement en fonction des paramètres de redirection du trafic définis pour le groupe de déploiement.

Autorisations IAM requises

Blue/green les déploiements sont rendus possibles par une combinaison d'Amazon ECS et CodeDeploy API. Les utilisateurs doivent disposer des autorisations appropriées pour ces services avant de pouvoir utiliser Amazon ECS blue/green déploiements dans Console de gestion AWS ou avec AWS CLI ou des kits de développement logiciel.

En plus des autorisations IAM standard nécessaires à la création et à la mise à jour de services, Amazon ECS exige les autorisations suivantes. Ces autorisations ont été ajoutées à la politique IAM `AmazonECS_FullAccess`. Pour plus d'informations, consultez Amazon ECS_FullAccess.

- `codedeploy:CreateApplication`
- `codedeploy:CreateDeployment`

- `codedeploy:CreateDeploymentGroup`
- `codedeploy:GetApplication`
- `codedeploy:GetDeployment`
- `codedeploy:GetDeploymentGroup`
- `codedeploy:ListApplications`
- `codedeploy:ListDeploymentGroups`
- `codedeploy:ListDeployments`
- `codedeploy:StopDeployment`
- `codedeploy:GetDeploymentTarget`
- `codedeploy:ListDeploymentTargets`
- `codedeploy:GetDeploymentConfig`
- `codedeploy:GetApplicationRevision`
- `codedeploy:RegisterApplicationRevision`
- `codedeploy:BatchGetApplicationRevisions`
- `codedeploy:BatchGetDeploymentGroups`
- `codedeploy:BatchGetDeployments`
- `codedeploy:BatchGetApplications`
- `codedeploy:ListApplicationRevisions`
- `codedeploy:ListDeploymentConfigs`
- `codedeploy:ContinueDeployment`
- `sns:ListTopics`
- `cloudwatch:DescribeAlarms`
- `lambda:ListFunctions`

> **Note**
>
> Outre les autorisations Amazon ECS standard nécessaires pour exécuter des tâches et des services, les utilisateurs doivent également disposer d'autorisations `iam:PassRole` pour pouvoir utiliser les rôles IAM dans le cadre des tâches.

CodeDeploy a besoin d'autorisations pour appeler les API Amazon ECS, modifier votre Elastic Load Balancing, invoquer les fonctions Lambda et décrire CloudWatch des alarmes, ainsi que des autorisations pour modifier le nombre souhaité de votre service en votre nom. Avant de créer un service Amazon ECS qui utilise le blue/green type de déploiement, vous devez créer un rôle IAM (`ecsCodeDeployRole`). Pour plus d'informations, consultez Amazon ECS CodeDeploy Rôle IAM.

Migrer CodeDeploy blue/green déploiements sur Amazon ECS blue/green déploiements

CodeDeploy blue/green et Amazon ECS blue/green les déploiements fournissent des fonctionnalités similaires, mais leur configuration et leur gestion diffèrent.

CodeDeploy blue/green vue d'ensemble du déploiement

Lors de la création d'un service Amazon ECS à l'aide de CodeDeploy, vous :

1. créez un équilibreur de charge avec un écouteur de production et (éventuellement) un écouteur de test. Chaque écouteur est configuré avec une seule règle (par défaut) qui achemine tout le trafic vers un seul groupe cible (le groupe cible principal) ;
2. créez un service Amazon ECS, configuré pour utiliser l'écouteur et le groupe cible, avec `deploymentController` type défini sur `CODE_DEPLOY`. La création d'un service entraîne la création d'un jeu de tâches (bleu) enregistré auprès du groupe cible spécifié ;
3. Créez un CodeDeploy groupe de déploiement (dans le cadre d'un CodeDeploy application), et configurez-le avec les détails du cluster Amazon ECS, le nom du service, les auditeurs de l'équilibreur de charge, deux groupes cibles (le groupe cible principal utilisé dans la règle d'écoute de production et un groupe cible secondaire à utiliser pour les tâches de remplacement), un rôle de service (pour accorder CodeDeploy autorisations pour manipuler les ressources Amazon ECS et Elastic Load Balancing) et divers paramètres qui contrôlent le comportement du déploiement.

Avec CodeDeploy, les nouvelles versions d'un service sont déployées à l'aide de `CreateDeployment()`, en spécifiant CodeDeploy nom de l'application, nom du groupe de déploiement et AppSpec fichier qui fournit des détails sur la nouvelle révision et les hooks de cycle de vie facultatifs. Le CodeDeploy deployment crée un ensemble de tâches de remplacement (vert) et enregistre ses tâches auprès du groupe cible secondaire. Lorsque celui-ci est opérationnel, il est disponible pour les tests (facultatif) et pour la production. Dans les deux cas, le réacheminement est réalisé en modifiant la règle d'écoute correspondante pour qu'elle pointe vers le groupe cible secondaire associé à le jeu de tâches vert. La restauration s'effectue en ramenant la règle d'écouteur de production au groupe cible principal.

Amazon ECS blue/green vue d'ensemble du déploiement

Avec Amazon ECS blue/green déploiements, La configuration de déploiement fait partie du service Amazon ECS lui-même :

1. Vous devez préconfigurer l'écouteur de production de l'équilibreur de charge avec une règle qui inclut deux groupes cibles avec des pondérations de 1 et 0.
2. Vous devez spécifier les ressources suivantes ou mettre à jour les ressources du service :
   - L'ARN de cette règle d'écoute
   - Les deux groupes cibles
   - Un rôle IAM autorisant Amazon ECS à appeler les API Elastic Load Balancing
   - Un rôle IAM facultatif pour exécuter les fonctions Lambda
   - Définissez le type `deploymentController` sur `ECS` et `deploymentConfiguration.strategy` sur `BLUE_GREEN`. Cela entraîne la création d'un déploiement de service (bleu) dont les tâches sont enregistrées auprès du groupe cible principal.

Avec Amazon ECS blue/green, une nouvelle révision de service est créée en appelant Amazon ECS `UpdateService()`, en transmettant les détails de la nouvelle révision. Le déploiement de service crée des tâches de révision de service (vertes) et les enregistre auprès du groupe cible secondaire. Amazon ECS gère les opérations de réacheminement et de restauration en modifiant les pondérations selon la règle d'écoute.

Principales différences de mise en œuvre

Bien que les deux approches aboutissent à la création d'un jeu de tâches initial, la mise en œuvre sous-jacente est différente :

- CodeDeploy utilise un ensemble de tâches, tandis qu'Amazon ECS utilise une révision de service. Les jeux de tâches Amazon ECS sont une ancienne structure qui a été remplacée par les révisions et les déploiements de service Amazon ECS. Ces derniers offrent une meilleure visibilité sur le processus de déploiement, ainsi que sur l'historique des déploiements et des révisions de service.
- Avec CodeDeploy, les hooks du cycle de vie sont spécifiés dans le cadre du AppSpec fichier qui est fourni à `CreateDeployment()`. Cela signifie que les hooks peuvent être modifiés d'un déploiement à l'autre. Avec Amazon ECS blue/green, les hooks sont spécifiés dans le cadre de la configuration du service, et toute mise à jour nécessiterait un `UpdateService()` appel.
- Les deux CodeDeploy et Amazon ECS blue/green utilisez Lambda pour l'implémentation des hooks, mais les entrées et sorties attendues diffèrent.

Avec CodeDeploy, la fonction doit appeler `PutLifecycleEventHookExecutionStatus()` pour renvoyer l'état du hook, qui peut être soit `SUCCEEDED` ou `FAILED`. Avec Amazon ECS, la réponse Lambda est utilisée pour indiquer l'état du hook.

- CodeDeploy invoque chaque hook comme un appel unique et s'attend à ce que l'état d'exécution final soit renvoyé dans l'heure. Les hooks Amazon ECS sont plus flexibles dans la mesure où ils peuvent renvoyer un indicateur `IN_PROGRESS`, qui spécifie que le hook doit être réinvoqué à plusieurs reprises jusqu'à ce qu'il aboutisse à `SUCCEEDED` ou `FAILED`. Pour plus d'informations, consultez Hooks de cycle de vie pour les déploiements de service Amazon ECS.

Approches de migration

Il existe trois approches principales pour migrer depuis CodeDeploy blue/green vers Amazon ECS blue/green déploiements. Chaque approche présente des caractéristiques différentes en termes de complexité, de risque, de capacité de restauration et de durée d'indisponibilité potentielle.

Réutilisez les mêmes ressources Elastic Load Balancing que celles utilisées pour CodeDeploy

Vous mettez à jour le service Amazon ECS existant pour utiliser le contrôleur de déploiement Amazon ECS avec blue/green stratégie de déploiement au lieu de CodeDeploy contrôleur de déploiement. Tenez compte des éléments suivants lorsque vous adoptez cette approche :

- La procédure de migration est plus simple, car vous mettez à jour le contrôleur de déploiement de service Amazon ECS et la stratégie de déploiement existants.
- Il n'y a aucune durée d'indisponibilité lorsque la configuration et la migration sont correctement effectuées.
- Une restauration nécessite que vous rétablissiez la révision de service.
- Le risque est élevé car il n'y a pas de parallèle blue/green configuration.

Vous utilisez le même équilibreur de charge, le même écouteur et les mêmes groupes cibles que ceux utilisés pour CodeDeploy. Si vous utilisez CloudFormation, voir Migrer un CloudFormation CodeDeploy blue/green modèle de déploiement sur un Amazon ECS blue/green CloudFormation modèle.

1. Modifiez la règle par défaut du production/test les auditeurs doivent inclure le groupe cible alternatif et définir le poids du groupe cible principal à 1 et du groupe cible alternatif à 0.

Pour CodeDeploy, les écouteurs de l'équilibreur de charge rattaché au service sont configurés avec une règle unique (par défaut) qui achemine tout le trafic vers un seul groupe cible. Pour Amazon ECS blue/green, les auditeurs de l'équilibreur de charge doivent être préconfigurés avec une règle qui inclut les deux groupes cibles avec des pondérations. Le groupe cible principal doit être pondéré à 1 et le groupe cible alternatif doit être pondéré à 0.

2. Mettez à jour le service Amazon ECS existant en appelant l'API `UpdateService` et en définissant le paramètre `deploymentController` sur `ECS` et le paramètre `deploymentStrategy` sur `BLUE_GREEN`. Spécifiez les ARN du groupe cible, du groupe cible alternatif, de l'écouteur de production et d'un écouteur de test facultatif.
3. Vérifiez que le service fonctionne comme prévu.
4. Supprimez le CodeDeploy configuration de ce service Amazon ECS car vous utilisez actuellement Amazon ECS blue/green.

Nouveau service avec équilibreur de charge existant

Cette approche utilise le blue/green stratégie de migration.

Tenez compte des éléments suivants lorsque vous adoptez cette approche :

- La perturbation est minime. Elle ne se produit que pendant le changement de port Elastic Load Balancing.
- Une restauration nécessite que vous rétablissiez l'échange de ports Elastic Load Balancing.
- Le risque est faible, car il existe des configurations parallèles. Vous pouvez donc effectuer un test avant le changement de trafic.

1. Laissez les auditeurs, les groupes cibles et le service Amazon ECS pour CodeDeploy configuration intacte afin que vous puissiez facilement revenir à cette configuration si nécessaire.
2. Créez des groupes cibles et des écouteurs (avec des ports différents de ceux des écouteurs d'origine) sous l'équilibreur de charge existant. Créez ensuite un service Amazon ECS qui correspond au service Amazon ECS existant, sauf que vous utilisez `ECS` comme contrôleur de déploiement, `BLUE_GREEN` comme stratégie de déploiement, et transmettez les ARN pour les nouveaux groupes cibles et les nouvelles règles d'écoute.
3. Vérifiez la nouvelle configuration en envoyant manuellement le trafic HTTP au service. Si tout se passe bien, échangez les ports des écouteurs d'origine et des nouveaux écouteurs pour acheminer le trafic vers la nouvelle configuration.

4. Vérifiez la nouvelle configuration, et si tout continue de fonctionner comme prévu, supprimez le CodeDeploy configuration.

Nouveau service avec un équilibreur de charge

Comme l'approche précédente, cette approche utilise le blue/green stratégie de migration. La principale différence réside dans le fait que le commutateur du CodeDeploy configuration sur Amazon ECS blue/green la configuration s'effectue au niveau d'une couche proxy inverse au-dessus de l'équilibreur de charge. Des exemples d'implémentations pour la couche proxy inverse sont Route 53 et CloudFront.

Cette approche convient aux clients qui disposent déjà de cette couche proxy inverse, et si toutes les communications avec le service passent par cette couche (par exemple, aucune communication directe au niveau de l’équilibreur de charge).

Tenez compte des éléments suivants lorsque vous adoptez cette approche :

- Elle nécessite une couche proxy inverse.
- La procédure de migration est plus complexe, car vous devez mettre à jour le contrôleur de déploiement de service Amazon ECS et la stratégie de déploiement existants.
- La perturbation est minime. Elle ne se produit que pendant le changement de port Elastic Load Balancing.
- Pour effectuer une restauration, vous devez annuler les modifications apportées à la configuration du proxy.
- Le risque est faible, car il existe des configurations parallèles. Vous pouvez donc effectuer un test avant le changement de trafic.

1. Ne modifiez pas l'existant CodeDeploy configuration intacte (équilibreur de charge, écouteurs, groupes cibles, service Amazon ECS, et CodeDeploy groupe de déploiement).
2. Créez un nouvel équilibreur de charge, des groupes cibles et des auditeurs configurés pour Amazon ECS blue/green déploiements.

   Configurez les ressources appropriées.

   - Application Load Balancer : pour plus d’informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.

- Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
- VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

3. Créez un service avec `ECS` en tant que contrôleur de déploiement et `BLUE_GREEN` en tant que stratégie de déploiement, en pointant vers les nouvelles ressources de l'équilibreur de charge.
4. Vérifiez la nouvelle configuration en la testant via le nouvel équilibreur de charge.
5. Mettez à jour la configuration du proxy inverse pour acheminer le trafic vers le nouvel équilibreur de charge.
6. Observez la nouvelle révision du service et si tout continue de fonctionner comme prévu, supprimez le CodeDeploy configuration.

Étapes suivantes

Après la migration vers Amazon ECS blue/green déploiements :

- Mettez à jour vos scripts de déploiement et CI/CD pipelines pour utiliser Amazon ECS `UpdateService` API au lieu de CodeDeploy `CreateDeployment` API.
- Mettez à jour votre système de surveillance et d'alerte pour suivre les déploiements de services Amazon ECS au lieu de CodeDeploy déploiements.
- Envisagez de mettre en œuvre des tests automatisés de votre nouveau processus de déploiement pour vous assurer qu'il fonctionne comme prévu.

Migrer depuis un CodeDeploy blue/green vers un Amazon ECS blue/green déploiement de services

En utilisant Amazon ECS blue/green lors de déploiements, vous pouvez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Vous devez créer de nouveaux hooks de cycle de vie pour votre Amazon ECS blue/green déploiement.

Conditions préalables

Effectuez les opérations suivantes avant de démarrer blue/green déploiement.

1. Remplacez l'Amazon ECS CodeDeploy Rôle IAM avec les autorisations suivantes.

- Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.
- Pour plus d'informations sur les autorisations Lambda, consultez la section Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

2. Éteindre CodeDeploy automatisation. Pour plus d'informations, voir Travailler avec des groupes de déploiement dans CodeDeploy dans le CodeDeploy Guide de l'utilisateur.
3. Assurez-vous de disposer des informations suivantes provenant de votre CodeDeploy blue/green déploiement. Vous pouvez réutiliser ces informations pour Amazon ECS blue/green déploiement :
   - Le groupe cible de production
   - L'écouteur de production
   - La règle de production
   - Le groupe cible de test.

     C'est le groupe cible de la révision de service verte.
4. Assurez-vous que les groupes cibles de votre Application Load Balancer sont correctement associés aux règles d'écoute :
   - Si vous n'utilisez pas d'écouteurs de test, les deux groupes cibles (production et test) doivent être associés aux règles d'écoute de production.
   - Si vous utilisez des écouteurs de test, un groupe cible doit être lié aux règles d'écoute de production et l'autre groupe cible doit être lié aux règles d'écoute de test.

   Si cette exigence n'est pas remplie, le déploiement de service échouera avec l'erreur suivante : `Service deployment rolled back because of invalid networking configuration. Both targetGroup and alternateTargetGroup must be associated with the productionListenerRule or testListenerRule.`
5. Vérifiez qu'aucun déploiement de service n'est en cours pour le service. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.
6. Amazon ECS blue/green les déploiements nécessitent que votre service utilise l'une des fonctionnalités suivantes : Configurez les ressources appropriées.
   - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
   - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.

- Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
- VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

7. Décidez si vous souhaitez exécuter des fonctions Lambda pour les étapes du cycle de vie pour les étapes d'Amazon ECS blue/green déploiement.
   - Avant augmentation horizontale
   - Après augmentation horizontale
   - Transfert du trafic test
   - Après transfert du trafic test
   - Transfert du trafic de production
   - Après transfert du trafic de production

   Créez des fonctions Lambda pour chaque étape du cycle de vie. Pour plus d'informations sur l'utilisation de Lambda, consultez la section Créer une fonction Lambda avec la console dans le Guide du développeur AWS Lambda .

Pour de plus amples informations sur la mise à jour du contrôleur de déploiement d'un service, consultez la section Mise à jour des paramètres d'un service Amazon ECS.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.

   La page des détails du cluster s'ouvre.
3. Depuis l'onglet Services, sélectionnez le service.

   La page des détails du service s'affiche.
4. Dans la bannière, choisissez Mettre à jour le type de contrôleur de déploiement.

   La page Procéder à la migration du type de contrôleur de déploiement s'affiche.
5. Développez Nouveau, puis spécifiez les paramètres suivants.
   a. Pour Type de contrôleur de déploiement, choisissez ECS.
   b. Pour Stratégie de déploiement, choisissez Blue/green.

c. Pour Durée de l'intégration, saisissez la durée pendant laquelle les révisions de service bleues et vertes s'exécutent.

d. Pour exécuter les fonctions Lambda pendant une étape du cycle de vie, sous Crochets liés au cycle de vie procédez comme suit pour chaque fonction Lambda unique :

i. Choisissez Ajouter.

Répétez l'opération pour chaque fonction que vous souhaitez exécuter.

ii. Pour Fonction Lambda, saisissez le nom de la fonction.

iii. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis à l'aide du blue/green autorisations.

Pour plus d'informations, consultez Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

iv. Pour Étapes du cycle de vie, sélectionnez les étapes exécutées par la fonction Lambda.

v. (Facultatif) Pour Détails du hook, saisissez une paire clé-valeur fournissant des informations sur le hook.

6. Développez Équilibrage de charge, puis configurez les éléments suivants :

a. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis à l'aide du blue/green autorisations.

Pour plus d'informations, consultez Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

b. Pour auditeur, choisissez l'auditeur de production parmi vos CodeDeploy blue/green déploiement.

c. Pour Règle de production, choisissez la règle de production dans votre CodeDeploy blue/green déploiement.

d. Pour Règle de test, choisissez la règle de test dans votre CodeDeploy blue/green déploiement.

e. Pour Groupe cible, choisissez le groupe cible de production parmi vos CodeDeploy blue/green déploiement.

f. Pour Groupe cible alternatif, choisissez le groupe cible du test parmi vos CodeDeploy blue/green déploiement.

7. Choisissez Mettre à jour.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il respecte les blue/green motif :
  - La révision de service verte est créée et augmentée horizontalement
  - Le trafic de test est acheminé vers la révision verte (si elle est configurée)
  - Le trafic de production est transféré vers la révision verte
  - Au terme de la durée de l'intégration, la révision bleue est résiliée

Migrer un CloudFormation CodeDeploy blue/green modèle de déploiement sur un Amazon ECS blue/green CloudFormation modèle

Migrer un CloudFormation modèle qui utilise CodeDeploy blue/green déploiements pour les services Amazon ECS vers un service utilisant l'Amazon ECS natif blue/green stratégie de déploiement. La migration fait suite à la procédure « Réutiliser les mêmes ressources Elastic Load Balancing que celles utilisées pour CodeDeploy« approche. Pour plus d'informations, consultez the section called "Migrer CodeDeploy blue/green déploiements sur Amazon ECS blue/green déploiements".

Modèle source.

Ce modèle utilise le `AWS::CodeDeployBlueGreen` transformer et `AWS::CodeDeploy::BlueGreen` crochet à implémenter blue/green déploiements pour un service Amazon ECS.

Source

C'est l'intégrale CloudFormation modèle utilisant CodeDeploy blue/green déploiement. Pour plus d'informations, voir Blue/green exemple de modèle de déploiement dans le AWS CloudFormation Guide de l'utilisateur:

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
```

```
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Transform": [
    "AWS::CodeDeployBlueGreen"
  ],
  "Hooks": {
    "CodeDeployBlueGreenHook": {
      "Type": "AWS::CodeDeploy::BlueGreen",
      "Properties": {
        "TrafficRoutingConfig": {
          "Type": "TimeBasedCanary",
          "TimeBasedCanary": {
            "StepPercentage": 15,
            "BakeTimeMins": 5
          }
        },
        "Applications": [
          {
            "Target": {
              "Type": "AWS::ECS::Service",
              "LogicalID": "ECSDemoService"
            },
            "ECSAttributes": {
              "TaskDefinitions": [
                "BlueTaskDefinition",
                "GreenTaskDefinition"
              ],
              "TaskSets": [
                "BlueTaskSet",
                "GreenTaskSet"
              ],
              "TrafficRouting": {
                "ProdTrafficRoute": {
                  "Type": "AWS::ElasticLoadBalancingV2::Listener",
                  "LogicalID": "ALBListenerProdTraffic"
                },
                "TargetGroups": [
                  "ALBTargetGroupBlue",
                  "ALBTargetGroupGreen"
                ]
              }
```

```
            }
          }
        ]
      }
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
          {
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 8080,
            "ToPort": 8080,
            "CidrIp": "0.0.0.0/0"
          },
          {
            "IpProtocol": "tcp",
            "FromPort": 22,
            "ToPort": 22,
            "CidrIp": "0.0.0.0/0"
          }
        ]
      }
    },
    "ALBTargetGroupBlue": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
```

```
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
```

```
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
```

```
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                }
              ]
            }
          }
        ],
        "Conditions": [
          {
            "Field": "http-header",
            "HttpHeaderConfig": {
              "HttpHeaderName": "User-Agent",
              "Values": [
                "Mozilla"
              ]
            }
          }
        ],
        "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
        "Priority": 1
      }
    },
    "ECSTaskExecutionRole": {
      "Type": "AWS::IAM::Role",
      "Properties": {
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs-tasks.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
        ]
```

```
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "EXTERNAL"
        }
      }
    },
    "BlueTaskSet": {
```

```
      "Type": "AWS::ECS::TaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "LaunchType": "FARGATE",
        "NetworkConfiguration": {
          "AwsVpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "PlatformVersion": "1.4.0",
        "Scale": {
          "Unit": "PERCENT",
          "Value": 100
        },
        "Service": {"Ref": "ECSDemoService"},
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"}
          }
        ]
      }
    },
    "PrimaryTaskSet": {
      "Type": "AWS::ECS::PrimaryTaskSet",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "Service": {"Ref": "ECSDemoService"},
        "TaskSetId": {"Fn::GetAtt": ["BlueTaskSet", "Id"]}
      }
    }
  }
}
```

Étapes de la migration

Supprimer CodeDeploy-specific ressources

Vous n'avez plus besoin des propriétés suivantes :

- La transformation `AWS::CodeDeployBlueGreen`
- Le hook `CodeDeployBlueGreenHook`
- Les ressources `GreenTaskDefinition` et `GreenTaskSet` (elles seront gérées par Amazon ECS)
- La ressource `PrimaryTaskSet` (Amazon ECS gérera les jeux de tâches en interne)

Reconfigurer l'écouteur de l'équilibreur de charge

Modifiez la ressource `ALBListenerProdTraffic` pour utiliser une action directe avec deux groupes cibles :

```
{
  "DefaultActions": [
    {
      "Type": "forward",
      "ForwardConfig": {
        "TargetGroups": [
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "Weight": 1
          },
          {
            "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
            "Weight": 0
          }
        ]
      }
    }
  ]
}
```

Mise à jour des propriétés de déploiement

Mettez à jour et ajoutez les éléments suivants :

- Modifiez la propriété `DeploymentController` en remplaçant `EXTERNAL` par `ECS`.
- Ajoutez la propriété `Strategy` et définissez-la sur BLUE_GREEN.
- Ajoutez de la propriété `BakeTimeInMinutes`.

```
{
  "DeploymentConfiguration": {
    "MaximumPercent": 200,
    "MinimumHealthyPercent": 100,
    "DeploymentCircuitBreaker": {
      "Enable": true,
      "Rollback": true
    },
    "BakeTimeInMinutes": 5,
    "Strategy": "BLUE_GREEN"
  }
}
```

- Ajoutez la configuration de l'équilibreur de charge au service :

```
{
  "LoadBalancers": [
    {
      "ContainerName": "DemoApp",
      "ContainerPort": 80,
      "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
      "AdvancedConfiguration": {
        "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
        "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
        "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers", "Arn"]}
      }
    }
  ]
}
```

- Ajoutez la référence de définition de tâche au service :

```
{
  "TaskDefinition": {"Ref": "BlueTaskDefinition"}
}
```

Créez le AmazonECSInfrastructureRolePolicyForLoadBalancers rôle

Ajoutez un nouveau rôle IAM qui permet à Amazon ECS de gérer les ressources de l'équilibreur de charge. Pour plus d'informations, consultez the section called “Rôle IAM d'infrastructure pour les équilibreurs de charge”

Recommandations de test

1. Déployez le modèle migré dans un environnement hors production.
2. Vérifiez que le service se déploie correctement avec la configuration initiale.
3. Testez un déploiement en mettant à jour la définition de la tâche et en respectant les blue/green processus de déploiement.
4. Vérifiez que le trafic est correctement transféré entre les déploiements bleu et vert.
5. Testez la fonctionnalité de restauration en forçant un échec de déploiement.

Modèle après migration

Modèle final

C'est l'intégrale CloudFormation modèle utilisant un Amazon ECS blue/green déploiement :

```
{
  "AWSTemplateFormatVersion": "2010-09-09",
  "Parameters": {
    "Vpc": {
      "Type": "AWS::EC2::VPC::Id"
    },
    "Subnet1": {
      "Type": "AWS::EC2::Subnet::Id"
    },
    "Subnet2": {
      "Type": "AWS::EC2::Subnet::Id"
    }
  },
  "Resources": {
    "ExampleSecurityGroup": {
      "Type": "AWS::EC2::SecurityGroup",
      "Properties": {
        "GroupDescription": "Security group for ec2 access",
        "VpcId": {"Ref": "Vpc"},
        "SecurityGroupIngress": [
```

```
        {
          "IpProtocol": "tcp",
          "FromPort": 80,
          "ToPort": 80,
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": 8080,
          "ToPort": 8080,
          "CidrIp": "0.0.0.0/0"
        },
        {
          "IpProtocol": "tcp",
          "FromPort": 22,
          "ToPort": 22,
          "CidrIp": "0.0.0.0/0"
        }
      ]
    }
  },
  "ALBTargetGroupBlue": {
    "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
    "Properties": {
      "HealthCheckIntervalSeconds": 5,
      "HealthCheckPath": "/",
      "HealthCheckPort": "80",
      "HealthCheckProtocol": "HTTP",
      "HealthCheckTimeoutSeconds": 2,
      "HealthyThresholdCount": 2,
      "Matcher": {
        "HttpCode": "200"
      },
      "Port": 80,
      "Protocol": "HTTP",
      "Tags": [
        {
          "Key": "Group",
          "Value": "Example"
        }
      ],
      "TargetType": "ip",
      "UnhealthyThresholdCount": 4,
      "VpcId": {"Ref": "Vpc"}
```

```
      }
    },
    "ALBTargetGroupGreen": {
      "Type": "AWS::ElasticLoadBalancingV2::TargetGroup",
      "Properties": {
        "HealthCheckIntervalSeconds": 5,
        "HealthCheckPath": "/",
        "HealthCheckPort": "80",
        "HealthCheckProtocol": "HTTP",
        "HealthCheckTimeoutSeconds": 2,
        "HealthyThresholdCount": 2,
        "Matcher": {
          "HttpCode": "200"
        },
        "Port": 80,
        "Protocol": "HTTP",
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
        "TargetType": "ip",
        "UnhealthyThresholdCount": 4,
        "VpcId": {"Ref": "Vpc"}
      }
    },
    "ExampleALB": {
      "Type": "AWS::ElasticLoadBalancingV2::LoadBalancer",
      "Properties": {
        "Scheme": "internet-facing",
        "SecurityGroups": [
          {"Ref": "ExampleSecurityGroup"}
        ],
        "Subnets": [
          {"Ref": "Subnet1"},
          {"Ref": "Subnet2"}
        ],
        "Tags": [
          {
            "Key": "Group",
            "Value": "Example"
          }
        ],
```

```
        "Type": "application",
        "IpAddressType": "ipv4"
      }
    },
    "ALBListenerProdTraffic": {
      "Type": "AWS::ElasticLoadBalancingV2::Listener",
      "Properties": {
        "DefaultActions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                },
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                  "Weight": 0
                }
              ]
            }
          }
        ],
        "LoadBalancerArn": {"Ref": "ExampleALB"},
        "Port": 80,
        "Protocol": "HTTP"
      }
    },
    "ALBListenerProdRule": {
      "Type": "AWS::ElasticLoadBalancingV2::ListenerRule",
      "Properties": {
        "Actions": [
          {
            "Type": "forward",
            "ForwardConfig": {
              "TargetGroups": [
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
                  "Weight": 1
                },
                {
                  "TargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
                  "Weight": 0
```

```
              }
            ]
          }
        }
      ],
      "Conditions": [
        {
          "Field": "http-header",
          "HttpHeaderConfig": {
            "HttpHeaderName": "User-Agent",
            "Values": [
              "Mozilla"
            ]
          }
        }
      ],
      "ListenerArn": {"Ref": "ALBListenerProdTraffic"},
      "Priority": 1
    }
  },
  "ECSTaskExecutionRole": {
    "Type": "AWS::IAM::Role",
    "Properties": {
      "AssumeRolePolicyDocument": {
        "Version": "2012-10-17",
        "Statement": [
          {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
              "Service": "ecs-tasks.amazonaws.com"
            },
            "Action": "sts:AssumeRole"
          }
        ]
      },
      "ManagedPolicyArns": [
        "arn:aws:iam::aws:policy/service-role/AmazonECSTaskExecutionRolePolicy"
      ]
    }
  },
  "ECSInfrastructureRoleForLoadBalancers": {
    "Type": "AWS::IAM::Role",
    "Properties": {
```

```
        "AssumeRolePolicyDocument": {
          "Version": "2012-10-17",
          "Statement": [
            {
              "Sid": "AllowAccessToECSForInfrastructureManagement",
              "Effect": "Allow",
              "Principal": {
                "Service": "ecs.amazonaws.com"
              },
              "Action": "sts:AssumeRole"
            }
          ]
        },
        "ManagedPolicyArns": [
          "arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForLoadBalancers"
        ]
      }
    },
    "BlueTaskDefinition": {
      "Type": "AWS::ECS::TaskDefinition",
      "Properties": {
        "ExecutionRoleArn": {"Fn::GetAtt": ["ECSTaskExecutionRole", "Arn"]},
        "ContainerDefinitions": [
          {
            "Name": "DemoApp",
            "Image": "nginxdemos/hello:latest",
            "Essential": true,
            "PortMappings": [
              {
                "HostPort": 80,
                "Protocol": "tcp",
                "ContainerPort": 80
              }
            ]
          }
        ],
        "RequiresCompatibilities": [
          "FARGATE"
        ],
        "NetworkMode": "awsvpc",
        "Cpu": "256",
        "Memory": "512",
        "Family": "ecs-demo"
      }
```

```
    },
    "ECSDemoCluster": {
      "Type": "AWS::ECS::Cluster",
      "Properties": {}
    },
    "ECSDemoService": {
      "Type": "AWS::ECS::Service",
      "Properties": {
        "Cluster": {"Ref": "ECSDemoCluster"},
        "DesiredCount": 1,
        "DeploymentController": {
          "Type": "ECS"
        },
        "DeploymentConfiguration": {
          "MaximumPercent": 200,
          "MinimumHealthyPercent": 100,
          "DeploymentCircuitBreaker": {
            "Enable": true,
            "Rollback": true
          },
          "BakeTimeInMinutes": 5,
          "Strategy": "BLUE_GREEN"
        },
        "NetworkConfiguration": {
          "AwsvpcConfiguration": {
            "AssignPublicIp": "ENABLED",
            "SecurityGroups": [
              {"Ref": "ExampleSecurityGroup"}
            ],
            "Subnets": [
              {"Ref": "Subnet1"},
              {"Ref": "Subnet2"}
            ]
          }
        },
        "LaunchType": "FARGATE",
        "PlatformVersion": "1.4.0",
        "TaskDefinition": {"Ref": "BlueTaskDefinition"},
        "LoadBalancers": [
          {
            "ContainerName": "DemoApp",
            "ContainerPort": 80,
            "TargetGroupArn": {"Ref": "ALBTargetGroupBlue"},
            "AdvancedConfiguration": {
```

```
              "AlternateTargetGroupArn": {"Ref": "ALBTargetGroupGreen"},
              "ProductionListenerRule": {"Ref": "ALBListenerProdRule"},
              "RoleArn": {"Fn::GetAtt": ["ECSInfrastructureRoleForLoadBalancers",
 "Arn"]}
            }
          }
        ]
      }
    }
  }
}
```

Migrer depuis un CodeDeploy blue/green à un déploiement du service de mise à jour continue d'Amazon ECS

Vous pouvez migrer vos déploiements de services depuis un CodeDeploy blue/green déploiement vers un déploiement de mises à jour continues d'Amazon ECS. Cela vous éloigne de CodeDeploy dépendance à l'utilisation d'un déploiement intégré.

Le planificateur de service Amazon ECS remplace les tâches en cours d'exécution par de nouvelles tâches. Le nombre de tâches ajoutées au service ou supprimées du service par Amazon ECS lors d'une mise à jour propagée est contrôlé par la configuration de déploiement du service.

Conditions préalables

Effectuez les opérations suivantes avant de démarrer blue/green déploiement.

1. Vous n'avez plus besoin d'Amazon ECS CodeDeploy Rôle IAM.
2. Éteindre CodeDeploy automatisation. Pour plus d'informations, voir Travailler avec des groupes de déploiement dans CodeDeploy dans le CodeDeploy Guide de l'utilisateur.
3. Vérifiez qu'aucun déploiement de service n'est en cours pour le service. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Pour de plus amples informations sur la mise à jour du contrôleur de déploiement d'un service, consultez la section Mise à jour des paramètres d'un service Amazon ECS.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.

La page des détails du cluster s'ouvre.

3. Depuis l'onglet Services, sélectionnez le service.

   La page des détails du service s'affiche.

4. Dans la bannière, choisissez Migrer.

   La page Mettre à jour la configuration du déploiement s'affiche.

5. Développez Options de déploiement, puis spécifiez les paramètres suivants.

   a. Pour Type de contrôleur de déploiement, choisissez ECS.

   b. Pour Stratégie de déploiement, choisissez Mise à jour propagée.

   c. Pour Min running tasks (Minimum de tâches en cours d'exécution), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configuration de déploiement).

   d. Pour Max running tasks (Maximum de tâches en cours d'exécution), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).

6. Développez Équilibrage de charge, puis configurez les éléments suivants :

   a. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis à l'aide du blue/green autorisations.

      Pour plus d'informations, consultez Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

   b. Pour auditeur, choisissez l'auditeur de production parmi vos CodeDeploy blue/green déploiement.

   c. Pour Groupe cible, choisissez le groupe cible de production parmi vos CodeDeploy blue/green déploiement.

7. Choisissez Mettre à jour.

Étapes suivantes

Vous devez mettre le service à jour pour que la modification prenne effet. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Mise à jour de la stratégie de déploiement d'Amazon ECS

Amazon ECS prend en charge plusieurs stratégies de déploiement pour mettre à jour vos services. Vous pouvez passer d'une stratégie à l'autre en fonction des exigences de votre application. Cette rubrique explique comment migrer entre des déploiements continus et blue/green déploiements.

Présentation des stratégies de déploiement d'Amazon ECS

Avant de passer d'une stratégie de déploiement à une autre, il est important de comprendre le fonctionnement de chaque stratégie et ses principales différences :

Déploiements continus

Dans le cadre d'un déploiement propagé, Amazon ECS remplace la version en cours d'exécution de votre application par une nouvelle version. Le planificateur de service utilise les paramètres de pourcentage minimal et maximal de bon fonctionnement pour déterminer la stratégie de déploiement.

Les déploiements propagés sont plus simples à configurer, mais offrent moins de contrôle sur le processus de déploiement et le routage du trafic.

Blue/green déploiements

Dans un blue/green lors du déploiement, Amazon ECS crée une nouvelle version de votre service (vert) en plus de la version existante (bleu). Cela vous permet de vérifier la nouvelle version avant d'y acheminer le trafic de production.

Blue/green les déploiements permettent de mieux contrôler le processus de déploiement, y compris les fonctionnalités de transfert du trafic, de test et de restauration.

Bonnes pratiques

Suivez ces pratiques exemplaires lors de la migration entre les stratégies de déploiement :

- Test dans un environnement hors production : testez toujours la mise à jour dans un environnement hors production avant d'appliquer des modifications aux services de production.

- Planification de la restauration : établissez un plan de restauration au cas où la mise à jour ne fonctionnerait pas comme prévu.
- Surveillance pendant la transition : surveillez de près votre service pendant et après la migration pour vous assurer qu'il continue de fonctionner correctement.
- Mise à jour de la documentation : mettez à jour votre documentation de déploiement pour refléter la nouvelle stratégie de déploiement.
- Considération de l'impact sur le trafic : déterminez l'impact de la mise à jour sur le trafic vers votre service et planifiez en conséquence.

Mise à jour de la stratégie de déploiement à partir d'une mise à jour continue vers Amazon ECS blue/green

Vous pouvez migrer d'un déploiement de mises à jour continues vers un Amazon ECS blue/green déploiement lorsque vous souhaitez apporter et tester des modifications de service avant de les implémenter dans un environnement de production.

Conditions préalables

Avant de migrer votre service de la version mobile vers blue/green déploiements, assurez-vous de disposer des éléments suivants :

- Attendez que tous les déploiements en cours soient terminés.
- Un service Amazon ECS existant utilisant la stratégie de déploiement propagé.
- Si plusieurs révisions de service reçoivent du trafic, Amazon ECS tente de consolider le trafic vers une seule révision pendant la migration. En cas d'échec, vous devrez peut-être mettre à jour manuellement votre service pour utiliser une seule révision avant de procéder à la migration.
- Configurez les autorisations appropriées.
  - Pour plus d'informations sur les autorisations Elastic Load Balancing, consultez la section Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge.
  - Pour plus d'informations sur les autorisations Lambda, consultez la section Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.
- Selon la configuration, vous devez effectuer l'une des actions suivantes :
  - Si votre service utilise Elastic Load Balancing, mettez-le à jour avec la nouvelle « AdvancedConfiguration » et lancez un déploiement propagé.
  - Si votre service utilise Service Connect, mettez-le à jour et lancez un déploiement propagé.

  - Si votre service utilise à la fois Elastic Load Balancing et Service Connect, effectuez les deux étapes précédentes (vous pouvez utiliser un UpdateService demande).
  - Si votre service n'en utilise aucune, aucune opération supplémentaire n'est nécessaire.
- Amazon ECS blue/green les déploiements nécessitent que votre service utilise l'une des fonctionnalités suivantes. Configurez les ressources appropriées.
  - Application Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge des applications pour blue/greendéploiements linéaires et Canary.
  - Network Load Balancer : pour plus d'informations, consultez la section Ressources sur l'équilibreur de charge réseau pour Amazon ECS blue/greendéploiements linéaires et canariens.
  - Service Connect : pour plus d'informations, consultez la section Ressources Service Connect pour Amazon ECS blue/greendéploiements linéaires et Canary.
  - VPC Lattice - Pour plus d'informations, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

Procédure

1. Ouvrez la console Amazon ECS à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, sélectionnez le cluster qui contient le service pour lequel vous souhaitez procéder à la migration.

   La page des détails du cluster s'affiche.
4. Sur la page Détails du cluster, choisissez l'onglet Services,
5. Choisissez le type de rôle Service , puis Mettre à jour.

   La page de mise à jour du service s'affiche
6. Développez Options de déploiement, puis procédez comme suit :
7. Pour Stratégie de déploiement, choisissez Blue/green.
8. Configurez le blue/green paramètres de déploiement :

   a. Pour Durée de l'intégration, saisissez le nombre de minutes pendant lesquelles les révisions de service bleues et vertes seront exécutées simultanément avant que la révision bleue ne soit résiliée.

      Cela laisse le temps de procéder à des vérifications et à des tests.

b. (Facultatif) Configurez les fonctions Lambda pour qu'elles s'exécutent à des étapes spécifiques du déploiement. Dans Hooks de cycle de vie du déploiement, configurez les fonctions Lambda pour les étapes suivantes :

- Avant augmentation horizontale : s'exécute avant d'augmenter verticalement la révision de service verte
- Après augmentation horizontale : s'exécute après avoir augmenté verticalement la révision de service verte
- Transfert du trafic test : s'exécute pendant le routage du trafic test vers la révision verte du service
- Après transfert du trafic test : s'exécute après que le trafic test a été acheminé vers la révision de service verte
- Transfert du trafic de production : s'exécute pendant le routage du trafic de production vers la révision de service vert
- Après transfert du trafic de production : s'exécute après que le trafic de production a été acheminé vers la révision de service vert

Pour ajouter un hook de cycle de vie

a. Choisissez Ajouter.
b. Pour Fonction Lambda, saisissez le nom ou l'ARN de la fonction.
c. Pour Rôle, choisissez le rôle IAM qui dispose de l'autorisation d'appeler la fonction Lambda.
d. Pour Étapes du cycle de vie, sélectionnez les étapes dans lesquelles la fonction Lambda doit être exécutée.
e. Facultatif : pour Détails du hook, saisissez des paires clé-valeur pour fournir des informations supplémentaires au hook.

9. Configurez les paramètres de l'équilibreur de charge :

a. Sous Équilibrage de charge, vérifiez que votre service est configuré pour utiliser un équilibreur de charge.
b. Pour Groupe cible, choisissez le groupe cible principal pour votre environnement de production (bleu).
c. Pour Groupe cible alternatif, choisissez le groupe cible pour votre environnement de test (vert).

d. Pour Règle d’écoute de production, choisissez la règle d’écoute pour acheminer le trafic de production.

e. Facultatif : pour Règle d’écoute de test, choisissez une règle d’écoute pour acheminer le trafic de test vers votre environnement vert.

f. Pour Rôle, choisissez le rôle IAM qui permet à Amazon ECS de gérer votre équilibreur de charge.

10. Examinez vos modifications, puis choisissez Mettre à jour.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour plus d’informations, consultez Mettre à jour un service Amazon ECS.
- Surveillez le processus de déploiement pour vous assurer qu'il respecte les blue/green motif :
  - La révision de service verte est créée et augmentée horizontalement
  - Le trafic de test est acheminé vers la révision verte (si elle est configurée)
  - Le trafic de production est transféré vers la révision verte
  - Au terme de la durée de l’intégration, la révision bleue est résiliée

Mise à jour de la stratégie de déploiement depuis Amazon ECS blue/green vers une mise à jour continue

Vous pouvez migrer un blue/green déploiement vers un déploiement de mises à jour continues.

Gardez les considérations suivantes à l’esprit lors de la migration vers des déploiements propagés :

- Gestion du trafic : avec les déploiements propagés, les nouvelles tâches commencent à recevoir du trafic dès qu’elles passent les surveillances de l’état. Il n'y a pas de phase de test distincte comme pour blue/green déploiements.
- Efficacité des ressources: Les déploiements continus utilisent généralement moins de ressources que blue/green déploiements car ils remplacent les tâches de manière incrémentielle au lieu de créer un environnement dupliqué complet.
- Complexité de la restauration: Les déploiements progressifs compliquent les annulations par rapport à blue/green déploiements. Si vous devez revenir en arrière, vous devez lancer un nouveau déploiement avec la définition de tâche précédente.

- Vitesse de déploiement: Les déploiements continus peuvent prendre plus de temps que blue/green déploiements, en particulier pour les services comportant de nombreuses tâches.
- Configuration de l'équilibreur de charge : votre configuration d'équilibreur de charge existante continuera de fonctionner avec des déploiements propagés, mais le comportement de transfert du trafic sera différent.

Conditions préalables

Avant de migrer votre service depuis blue/green pour les déploiements continus, assurez-vous de disposer des éléments suivants :

- Un service Amazon ECS existant utilisant le blue/green stratégie de déploiement
- Aucun déploiement en cours pour le service (attendez que les déploiements en cours soient terminés)
- Une compréhension claire de la façon dont votre service se comportera lors de déploiements propagés

> **Note**
>
> Vous ne pouvez pas procéder à la migration d'un service vers un déploiement propagé s'il est en cours de déploiement. Attendez que tous les déploiements en cours soient terminés avant de continuer.

Procédure de migration

Suivez ces étapes pour migrer votre service Amazon ECS depuis blue/green aux déploiements continus :

1. Ouvrez la console Amazon ECS à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, sélectionnez le cluster qui contient le service pour lequel vous souhaitez procéder à la migration.
4. Sur la page Détails du cluster, choisissez l'onglet Services,
5. Sélectionnez le service pour lequel vous souhaitez procéder à la migration, puis choisissez Mettre à jour.

6. Sur la page Mettre à jour le service, accédez à la section Options de déploiement et développez-la si nécessaire.
7. Pour Stratégie de déploiement, choisissez Mise à jour propagée.
8. Configurez les paramètres de déploiement propagé :
    a. Pour Pourcentage minimum sain, saisissez une limite inférieure pour le nombre de tâches pour lesquelles votre service doit rester dans l'état `RUNNING` lors d'un déploiement. Cette valeur est exprimée en pourcentage du nombre de tâches souhaité pour le service.
    b. Dans Pourcentage maximal, saisissez le pourcentage maximal de tâches autorisées à l'état `RUNNING` ou `PENDING` lors d'un déploiement. Cette valeur est exprimée en pourcentage du nombre de tâches souhaité pour le service.
9. Facultatif : sous Détection des échecs de déploiement, configurez la manière dont Amazon ECS détecte et gère les échecs de déploiement :
    a. Pour activer le disjoncteur de déploiement, sélectionnez Utiliser le disjoncteur de déploiement.
    b. Pour restaurer automatiquement les déploiements ayant échoué, choisissez Restaurer en cas d'échec.
10. Passez en revue vos modifications de configuration, puis choisissez Mettre à jour pour enregistrer vos modifications et procéder à la migration du service vers un déploiement propagé.

Amazon ECS mettra à jour la configuration de votre service afin d'utiliser la stratégie de déploiement propagé. La prochaine fois que vous mettrez à jour votre service, celui-ci utilisera le processus de déploiement propagé.

> **Note**
>
> Lorsque vous migrez depuis blue/green vers un déploiement continu, Amazon ECS gère la transition en :
>
> 1. identifiant la révision de service active actuelle qui reçoit le trafic ;
> 2. maintenant la configuration existante de l'équilibreur de charge, mais en modifiant la façon dont les nouveaux déploiements sont gérés ;
> 3. préparant la service pour les futurs déploiements continus.

Étapes suivantes

- Mettez à jour le service pour démarrer le déploiement. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Résolution des problèmes liés aux mises à jour de la stratégie de déploiement Amazon ECS

Cette section fournit des solutions aux problèmes courants que vous pouvez rencontrer lors de la migration de stratégies de déploiement.

Révisions de services ou jeux de tâches multiples

Les problèmes suivants concernent le fait d'avoir plusieurs révisions de service pour un déploiement.

Jeux de tâches multiples lors de la mise à jour du contrôleur de déploiement ECS

> Message d'erreur : `Updating the deployment controller is not supported when there are multiple tasksets in the service. Please ensure your service has only one taskset and try again.`
>
> Solution : cette erreur se produit lorsque vous tentez de modifier le type de contrôleur de déploiement d'un service comportant plusieurs jeux de tâches actifs. Pour résoudre ce problème pour le contrôleur de déploiement `CODE_DEPLOY` ou `EXTERNAL` :
>
> 1. Vérifiez les jeux de tâches actuels :
>
> ```
> aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets"
> ```
>
> 2. Attendez que tous les déploiements en cours soient terminés.
> 3. Forcez un nouveau déploiement pour nettoyer les jeux de tâches :
>
> ```
> aws ecs update-service --cluster your-cluster-name --service your-service-name
>  --force-new-deployment
> ```
>
> 4. Si nécessaire, supprimez manuellement des jeux de tâches supplémentaires :
>
> ```
> aws ecs delete-task-set --cluster your-cluster-name --service your-service-name
>  --task-set task-set-id
> ```

5. Lorsqu'il ne reste qu'un seul jeu de tâches, réessayez de mettre à jour le contrôleur de déploiement.

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Jeu de tâches principal manquant lors de la mise à jour du contrôleur de déploiement ECS

Message d'erreur : `Updating the deployment controller requires a primary taskset in the service. Please ensure your service has a primary taskset and try again.`

Solution : cette erreur se produit lorsque vous tentez de modifier le type de contrôleur de déploiement d'un service qui ne dispose pas d'un jeu de tâches principal. Pour résoudre ce problème :

1. Vérifiez l'état du service et les jeux de tâches.). Si un jeu de tâches existe dans le service, il doit être marqué comme `ACTIVE`.

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].taskSets[*].[status,id]
```

   S'il n'y a aucun jeu de tâches dans l'état `ACTIVE`, procédez à la migration du déploiement. Pour plus d'informations, consultez la section Approches de migration.

2. Si aucune tâche n'est en cours d'exécution, déployez au moins une tâche en mettant à jour le service :

```
aws ecs update-service-primary-task-set --cluster your-cluster-name --service your-service-name --primary-task-set your-taskset-id
```

   Cela marquera la tâche (auparavant `ACTIVE`) définie dans le service comme ayant le statut `PRIMARY`.

3. Attendez que la tâche atteigne un état d'exécution stable. Vous pouvez vérifier l'état en utilisant :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deployments"
```

4. Une fois que le service dispose d'un jeu de tâches principales avec des tâches en cours d'exécution, réessayez de mettre à jour le contrôleur de déploiement.

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Incompatibilité entre le type de détection des échecs de déploiement et le contrôleur de déploiement

Les problèmes suivants sont liés à une incompatibilité entre le type de détection des échecs de déploiement et le contrôleur de déploiement.

Disjoncteur de déploiement avec contrôleur non-ECS

Message d'erreur : `Deployment circuit breaker feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solution : cette erreur se produit lorsque vous tentez d'activer la fonctionnalité de disjoncteur de déploiement sur un service qui n'utilise pas le contrôleur de déploiement ECS. Le disjoncteur de déploiement n'est compatible qu'avec le contrôleur de déploiement ECS.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur de déploiement ECS :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur de déploiement ECS, activez le disjoncteur de déploiement :

```
aws ecs update-service --cluster your-cluster-name --
service your-service-name --deployment-configuration
 "deploymentCircuitBreaker={enable=true,rollback=true}"
```

Pour plus d'informations, consultez the section called " Comment le disjoncteur de déploiement détecte les pannes".

Alarm-based restauration avec un contrôleur non ECS

Message d'erreur : `Alarm based rollback feature is only supported with ECS deployment controller. Update to ECS deployment controller and try again.`

Solution : cette erreur se produit lors de la tentative de configuration de la restauration basée sur les alarmes sur un service qui n'utilise pas le contrôleur de déploiement ECS. La fonctionnalité de restauration basée sur les alarmes n'est compatible qu'avec le contrôleur de déploiement ECS.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur de déploiement ECS :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur de déploiement ECS, configurez la restauration basée sur les alarmes :

```
aws ecs update-service --cluster your-cluster-name --services your-service-name --deployment-configuration "alarms={alarmNames=[your-alarm-name],enable=true,rollback=true}"
```

Pour plus d'informations, consultez the section called "Comment CloudWatch les alarmes détectent les échecs de déploiement".

Inadéquation entre Service Connect et le contrôleur de déploiement

Les problèmes suivants sont liés à une inadéquation entre Service Connect et le contrôleur de déploiement.

Contrôleur EXTERNAL avec Service Connect

Message d'erreur : `The EXTERNAL deployment controller type is not supported for services using Service Connect.`

Solution : cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de déploiement EXTERNAL avec un service sur lequel Service Connect est activé. Le contrôleur EXTERNAL n'est pas compatible avec Service Connect.

1. Vérifiez si Service Connect est activé sur votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].serviceConnectConfiguration"
```

2. Si vous devez utiliser le contrôleur de déploiement EXTERNAL, désactivez Service Connect en mettant à jour votre service :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "{}"
```

3. Sinon, si vous devez utiliser Service Connect, utilisez plutôt le contrôleur de déploiement ECS :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Service Connect avec un contrôleur non-ECS

Message d'erreur : `Service Connect feature is only supported with ECS (rolling update) deployment controller. Update to ECS deployment controller and try again.`

Solution : cette erreur se produit lorsque vous tentez d'activer Service Connect sur un ordinateur qui n'utilise pas le contrôleur de déploiement ECS. La fonctionnalité Service Connect n'est compatible qu'avec le contrôleur de déploiement ECS.

1. Vérifiez le contrôleur de déploiement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].deploymentController"
```

2. Mettez à jour votre service pour utiliser le contrôleur de déploiement ECS :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

3. Une fois que le service utilise le contrôleur de déploiement ECS, activez Service Connect :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-connect-configuration "enabled=true,namespace=your-namespace"
```

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Inadéquation entre le type de contrôleur et la stratégie de planification

Les problèmes suivants concernent une inadéquation entre le type de contrôleur et la stratégie de planification.

Contrôleur `CODE_DEPLOY` avec stratégie de planification `DAEMON`

Message d'erreur : `The CODE_DEPLOY deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solution : cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de déploiement CODE_DEPLOY avec un service utilisant la stratégie de planification `DAEMON`. Le contrôleur `CODE_DEPLOY` n'est compatible qu'avec la stratégie de planification `REPLICA`.

1. Vérifiez la stratégie de planification actuelle de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].schedulingStrategy"
```

2. Si vous avez besoin blue/green déploiements, modifiez votre service pour utiliser le `REPLICA` stratégie de planification :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Sinon, si vous devez utiliser la stratégie de planification `DAEMON`, utilisez plutôt le contrôleur de déploiement `ECS` :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Contrôleur EXTERNE avec stratégie de planification DÉMON

Message d'erreur : `The EXTERNAL deployment controller type is not supported for services using the DAEMON scheduling strategy.`

Solution : cette erreur se produit lorsque vous tentez d'utiliser le contrôleur de déploiement EXTERNE avec un service ECS utilisant la stratégie de planification DÉMON. Le contrôleur EXTERNAL n'est compatible qu'avec la stratégie de planification REPLICA.

1. Vérifiez la stratégie de planification actuelle de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-
name --query "services[0].schedulingStrategy"
```

2. Si vous devez utiliser le contrôleur de déploiement `EXTERNAL`, modifiez votre service pour utiliser la stratégie de planification `REPLICA` :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --scheduling-strategy REPLICA
```

3. Sinon, si vous devez utiliser la stratégie de planification `DAEMON`, utilisez plutôt le contrôleur de déploiement `ECS` :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --deployment-controller type=ECS
```

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Enregistrements de services avec type de lancement externe

Message d'erreur : `Service registries are not supported for external launch type.`

Solution : cette erreur se produit lors de la tentative de configuration de la découverte de service (registres de services) pour un service utilisant le type de lancement `EXTERNAL`. La découverte de service n'est pas compatible avec le type de lancement `EXTERNAL`.

1. Vérifiez le type de lancement actuel de votre service :

```
aws ecs describe-services --cluster your-cluster-name --services your-service-name --query "services[0].launchType"
```

2. Si vous avez besoin de la découverte de service, modifiez votre service pour utiliser le type de lancement `EC2` ou `FARGATE` :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --launch-type FARGATE
```

3. Sinon, si vous devez utiliser le type de lancement `EXTERNAL`, supprimez la configuration du registre du service :

```
aws ecs update-service --cluster your-cluster-name --service your-service-name
 --service-registries "[]"
```

Pour plus d'informations, consultez the section called "Contrôleurs et stratégies de déploiement de service".

Rétablissement d'une mise à jour du contrôleur de déploiement

Si vous décidez de revenir au contrôleur de déploiement précédent, vous pouvez effectuer l'une des opérations suivantes :

- Si vous avez utilisé CloudFormation, vous pouvez utiliser le modèle précédent pour créer une nouvelle pile. Pour plus d'informations, consultez la section Créer une pile sous dans le Guide de l'utilisateur CloudFormation .

- Si vous avez utilisé la console Amazon ECS ou AWS CLI, vous pouvez mettre à jour le service. Pour plus d'informations, consultez the section called "Mise à jour d'un service".

  Si vous utilisez la commande update-service, utilisez l'option `--deployment-controller` et définissez-la sur le contrôleur de déploiement précédent.

Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS

Les déploiements de service fournissent une vue complète de vos déploiements. Les déploiements de service fournissent les informations suivantes sur le service :

- Configuration de la charge de travail actuellement déployée (révision de service source)
- Configuration de la charge de travail en cours de déploiement (la révision de service cible)
- Statut du déploiement
- Nombre de tâches ayant échoué détectées par le disjoncteur
- Le CloudWatch alarmes en état d'alarme
- Quand le déploiement de service a commencé et s'est terminé
- Détails d'une restauration, le cas échéant

Pour de plus amples informations sur les propriétés de déploiement de service, consultez la section Propriétés incluses dans un déploiement de service Amazon ECS.

Les déploiements de service sont en lecture seule et chacun possède un identifiant unique.

Il existe trois étapes de déploiement de service :

Étape	Définition	États associés
En attente	Un déploiement de service a été créé, mais n'a pas démarré.	PENDING
En continu	Le déploiement de service est en cours.	• EN_COURS • STOP_REQUESTED

Étape	Définition	États associés
		• ROLLBACK_REQUESTED • ROLLBACK_IN_PROGRESS
Terminé	Le déploiement de service est terminé (avec ou sans succès)	• SUCCESSFUL • STOPPED • RESTAURATION_RÉUSSIE • ROLLBACK_FAILED

Vous utilisez les déploiements de service pour comprendre le cycle de vie de votre service et pour déterminer si des mesures doivent être prises. Par exemple, en cas de restauration, vous devrez peut-être étudier le déploiement de service et les événements du service.

Vous pouvez consulter l'historique le plus récent des 90 jours des déploiements créés le 25 octobre 2024 ou après cette date à l'aide de la console, de l'API et de l' AWS CLI.

Vous pouvez arrêter un déploiement qui n'est pas terminé. Pour plus d'informations, consultez Arrêt de déploiements de service Amazon ECS.

Si un déploiement utilise des critères de réussite précoce, `DescribeServiceDeployments` renvoie les critères de succès initiaux configurés, et le nombre de tâches en cours qu'il indique est un instantané de la fin du déploiement plutôt que des chiffres en temps réel. Les déploiements qui se terminent en avance sont inclus lorsque vous filtrez `ListServiceDeployments` par un statut de `SUCCESSFUL`. Pour plus d'informations, consultez Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces.

Cycle de vie de déploiement de service

Amazon ECS crée automatiquement un déploiement de service lorsque l'une des actions suivantes se produit :

- Un utilisateur crée un service.
- Un utilisateur met à jour le service et utilise l'option « Forcer un nouveau déploiement ».
- Un utilisateur met à jour une ou plusieurs propriétés du service qui nécessitent un déploiement.

Pendant qu'un déploiement est en cours, Amazon ECS met à jour les propriétés de déploiement de service suivantes afin de refléter la progression du déploiement de service :

- L'état
- Le nombre de tâches en cours

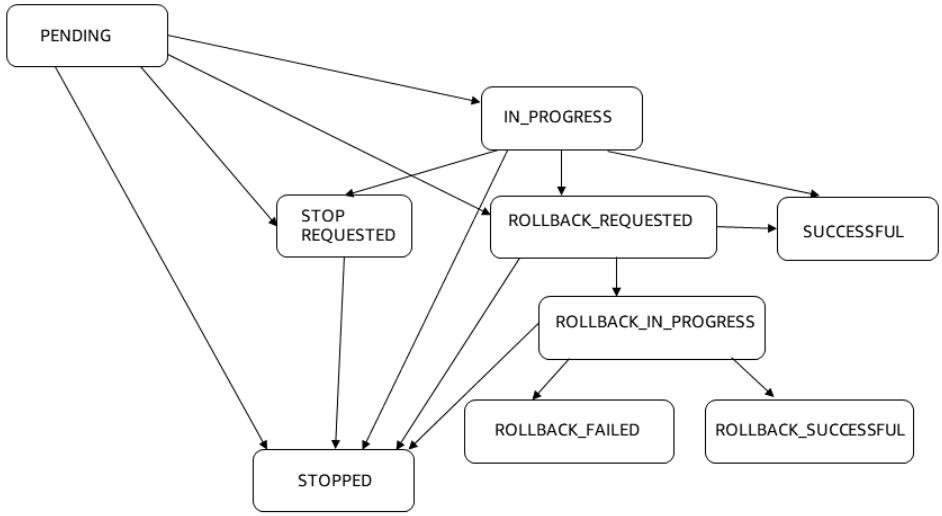
  Le nombre de tâches en cours indiqué dans la révision de service peut ne pas être égal au nombre réel de tâches en cours d'exécution. Ce nombre représente le nombre de tâches exécutées une fois le déploiement terminé. Par exemple, si vous avez lancé des tâches indépendamment du déploiement de service, ces tâches ne sont pas incluses dans le nombre de tâches en cours pour la révision de service.
- Détection des échecs du disjoncteur :
  - Nombre de tâches qui n'ont pas pu démarrer
- CloudWatch détection de panne d'alarme
  - Les alarmes actives
- Informations de restauration :
  - L'heure de début
  - La raison de la restauration
  - L'ARN de la révision de service utilisée pour la restauration
- La raison du statut

Amazon ECS supprime le déploiement de service lorsque vous supprimez un service.

États de déploiement de service

Le déploiement de service démarre dans l'état `PENDING`.

L'illustration suivante montre les états de déploiement de service qui peuvent se produire après l'état `PENDING` : `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_IN_PROGRESSS`, `ROLLBACK_FAILED`, `ROLLBACK_SUCCESSFUL` et `STOPPED`.



Les informations suivantes fournissent des informations détaillées sur les états de déploiement de service :

- `PENDING` : un déploiement de service a été créé, mais n'a pas démarré

  L'état peut se passer à `IN_PROGRESS`, `ROLLBACK_REQUESTED`, `STOP_REQUESTED` ou `STOPPED`.
- `IN_PROGRESS` : le déploiement de service est en cours.

  L'état peut passer à `SUCCESSFUL`, `STOP_REQUESTED`, `ROLLBACK_REQUESTED`, `ROLLBACK_IN_PROGRESS` et `STOPPED`.
- `STOP_REQUESTED` : l'état du déploiement de service passe à `STOP_REQUESTED` lorsque l'une des situations suivantes se produit :
  - Un utilisateur lance le déploiement d'un nouveau service.
  - L'option de restauration n'est pas utilisée pour le mécanisme de détection des défaillances (disjoncteur ou alarme) et le service n'atteint pas l'état `SUCCESSFUL`.

  L'état passe à `STOPPED`.
- `ROLLBACK_REQUESTED` : l'état du déploiement de service passe à `ROLLBACK_REQUESTED` lorsqu'un utilisateur requête une restauration par le biais de la console, de l'API ou de la CLI.

L'état peut passer à `SUCCESSFUL`, `ROLLBACK_IN_PROGRESS` et `STOPPED`.

- `SUCCESSFUL` : l'état du déploiement de service passe à `SUCCESSFUL` lorsque le déploiement de service est terminé avec succès.
- `ROLLBACK_IN_PROGRESS` : l'état du déploiement de service passe à `ROLLBACK_IN_PROGRESS` lorsque l'option de restauration est utilisée pour le mécanisme de détection des échecs (disjoncteur ou basé sur les alarmes) et que le service échoue.

  L'état passe à `ROLLBACK_SUCCESSFUL` ou `ROLLBACK_FAILED`.

Propriétés incluses dans un déploiement de service Amazon ECS

Les propriétés suivantes sont incluses dans un déploiement de service.

Propriété	Description
ARu de déploiement de service	L'ARN du déploiement de service.
ARN du service	L'ARN du service pour ce déploiement de service.
ARN de cluster	L'ARN du cluster qui héberge le service.
Heure de création du déploiement de service	L'heure à laquelle le déploiement de service a été créé.
Heure de début du déploiement de service	L'heure à laquelle le déploiement de service a commencé.
Heure de fin du déploiement de service	L'heure à laquelle le déploiement de service s'est terminé.

Propriété	Description
Heure d’arrêt du déploiement de service	L’heure à laquelle le déploiement de service s’est arrêté.
Heure de mise à jour du déploiement de service	L’heure de la dernière mise à jour du déploiement de service.
Révisions de service source	Les révisions de service actuellement en cours. Pour de plus amples informations sur les propriétés incluses, consultez la section Propriétés incluses dans une révision de service Amazon ECS.

Propriété	Description	
Configuration de déploiement	Les paramètres de déploiement, y compris la configuration du disjoncteur et les alarmes qui le déterminent. • La limite inférieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée, exprimée en pourcentage du nombre de tâches souhaité pour le service. • La limite supérieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée, exprimée en pourcentage du nombre de tâches souhaité pour le service. • La configuration du disjoncteur. • Les alarmes utilisées pour détecter les échecs.	

Propriété	Description
Révision du service cible	La révision de service à déployer. Une fois le déploiement terminé avec succès, la révision de service cible est la révision de service en cours.
État du déploiement de service	L'état du déploiement de service. Les valeurs valides sont EN_ATTENTE, SUCCÈS, ARRETÉ, ARRÊT_DEMANDÉ, ARRÊT_EN_COURS, EN_COURS, RESTAURATION_EN_COURS, RESTAURATION_RÉUSSIE et RESTAURATION_ÉCHOUÉE.
Informations sur les statuts de déploiement des services	Informations sur la raison pour laquelle le déploiement de service se trouve dans l'état actuel. Par exemple, le disjoncteur a détecté un échec.
Informations de restauration	Les options de restauration utilisées par le déploiement de service en cas d'échec du déploiement.
Options de disjoncteur de déploiement de service	Le disjoncteur qui détermine que le déploiement d'un service a échoué.

Propriété	Description	
CloudWatch alarmes pour le déploiement du service	Le CloudWatch des alarmes qui déterminent l'échec du déploiement d'un service.	

Autorisations requises pour consulter les déploiements de service Amazon ECS

Lorsque vous suivez la pratique exemplaire consistant à octroyer le moindre privilège, vous devez ajouter des autorisations supplémentaires afin de visualiser les déploiements de service dans la console.

Vous devez avoir accès aux actions suivantes :

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Vous devez avoir accès aux ressources suivantes :

- Service
- Déploiements de service
- Révision de service

L'exemple de politique suivant contient les autorisations requises et limite les actions à un service spécifié.

Remplacez `account`, `cluster-name` et `service-name` par vos propres valeurs.

JSON

```
{
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
```

```
            "ecs:ListServiceDeployments",
            "ecs:DescribeServiceDeployments",
            "ecs:DescribeServiceRevisions"
        ],
        "Resource": [
            "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-
name",
            "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
            "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/
service-name/*"
            ]
        }
   ]
}
```

Affichage des déploiements de service Amazon ECS

Vous pouvez consulter l'historique des 90 derniers jours pour les déploiements créés à partir du 25 octobre 2024. Les déploiements de service peuvent avoir l'un des états suivants :

- In-progress
- En attente
- Terminé

Vous pouvez utiliser ces informations pour déterminer si vous devez mettre à jour le mode de déploiement du service ou de la révision de service. Pour de plus amples informations sur les propriétés incluses, consultez la section Propriétés incluses dans un déploiement de service Amazon ECS.

Avant de commencer, configurez les autorisations requises pour consulter les déploiements de service. Pour plus d'informations, consultez Autorisations requises pour consulter les déploiements de service Amazon ECS.

Amazon ECS Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

La page des détails du service s'affiche.

4. Sur la page des détails du service, choisissez Déploiements.
5. Choisissez le déploiement de service à afficher.

Pour afficher les déploiements de service pour ce type de déploiement	Faites ceci
Déploiement en cours	Sous Déploiements en cours, choisissez l'ID de déploiement.
Dernier déploiement	Sous Dernier déploiement, choisissez l'ID de déploiement.
Déploiements terminés	Sous Déploiements de service, choisissez l'ID de déploiement.

La page de détails du déploiement de service s'affiche.

6. (Facultatif) Comparez les révisions de service pour voir les différences.

   Sous Révisions de service, choisissez Comparer les révisions, puis sélectionnez deux révisions à comparer.

   Les révisions de service sont affichées côte à côte avec les différences mises en évidence.

AWS CLI

1. Exécutez `list-service-deployments` pour récupérer l'ARN de déploiement de service.

   Remplacez les variables par vos valeurs.

```
aws ecs list-service-deployments --cluster cluster-name --service service-name
```

Notez le serviceDeploymentArn pour le déploiement que vous souhaitez consulter.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "targetServiceRevisionArn": "arn:aws:ecs:us-
west-2:123456789012:service-revision/example/sd-example/4980306466373577095",
            "status": "SUCCESSFUL"
        }
    ]
}
```

2. Exécutez `describe-service-deployments`. Utilisez celui `serviceDeploymentArn` qui a été renvoyé par `list-service-deployments`.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-deployments --service-deployment-arns
 arn:aws:ecs:region:123456789012:service-deployment/cluster-name/service-
name/NCWGC2ZR-taawPAYrIaU5
```

Étapes suivantes

Vous pouvez afficher les détails des révisions de service lors du déploiement. Pour plus d'informations, consultez Affichage des détails de la révision d'Amazon ECS

Révisions de service Amazon ECS

Une révision de service contient un enregistrement de la configuration de la charge de travail qu'Amazon ECS tente de déployer. Chaque fois que vous créez ou déployez un service, Amazon ECS crée et capture automatiquement la configuration que vous essayez de déployer dans la révision de service.

Les révisions de service sont en lecture seule et possèdent des identifiants uniques. Pour de plus amples informations sur les propriétés incluses, consultez la section Propriétés incluses dans une révision de service Amazon ECS.

Les révisions de service offrent les avantages suivants :

- Lors d'un déploiement de service, vous pouvez comparer la révision de service actuellement déployée (source) avec celle en cours de déploiement (cible).
- Lorsque vous utilisez l'option de restauration pour un déploiement de service, Amazon ECS restaure automatiquement le déploiement de service jusqu'à la dernière révision de service déployée avec succès.
- Les révisions de service contiennent l'enregistrement de la configuration de la charge de travail dans une ressource.

Cycle de vie de la révision de service

Amazon ECS crée automatiquement une révision de service lorsque vous créez un service ou que vous mettez à jour une propriété de service qui lance un déploiement.

Amazon ECS ne crée pas de révision de service pour une opération de restauration. Amazon ECS utilise la dernière révision de service réussie pour la restauration.

La révision de service est immuable.

Amazon ECS supprime la révision de service lorsque vous supprimez un service.

Vous pouvez consulter les révisions de service créées le 25 octobre 2024 ou après cette date à l'aide de la console, de l'API et de la CLI.

Propriétés incluses dans une révision de service Amazon ECS

Les propriétés suivantes sont incluses dans une révision de service.

Ressource	Description	
ARN du service	ARN qui identifie le service.	
ARN de cluster	L'ARN du cluster qui héberge le service.	

Ressource	Description
ARN de définition de tâche	L'ARN de la définition de tâche utilisée pour les tâches de service.
Enregistrements de service	Les détails des registres de service utilisés pour la découverte de service. • Nom du conteneur La valeur du nom du conteneur à utiliser pour votre service de découverte de service. • Port du conteneur La valeur du port à utiliser pour votre service de découverte de service. Cette valeur est également spécifiée dans la définition de tâche. • Le port Valeur de port utilisée si votre service de découvert e de service a spécifié un registre SRV. • ARN du registre L'Amazon Resource Name (ARN) du registre de service.

Ressource	Description
Fournisseurs de capacité	Les détails de la stratégie de fournisseur de capacité. • Nom du fournisseur de capacité Le nom du fournisseur de capacité utilisé pour le service. • Poids du fournisseur de capacité Le pourcentage relatif du nombre total de tâches lancées qui devraient utiliser le fournisseur de capacité spécifié. • Base des fournisseurs de capacité Le nombre minimal de tâches à exécuter sur le fournisseur de capacité spécifié.
Images de conteneurs	Les détails concernant les images conteneur. • Nom du conteneur • Condensé de l’image • Image de conteneur

Ressource	Description
Mise en réseau	Configuration réseau du service. • Existence d’une adresse IP publique • Sous-réseaux dans lesquels les tâches s’exécutent • Groupes de sécurité contrôlant le trafic du service
Type de lancement	L’option de calcul utilisée pour le service.

Fargate-specific propriétés	Lorsque vous utilisez Fargate, il s’agit d’informations sur la version de Fargate.
	• La version de plateforme et la famille de plateforme
	• Stockage éphémère
	  • La quantité de stockage éphémère à allouer pour le déploiement.
	  • L’ID de la clé KMS à utiliser pour le chiffrement par défaut.

Ressource	Description
Volumes Amazon EBS configurés lors du déploiement	La configuration d'un volume spécifié dans la définition de tâche en tant que volume configuré au moment du lancement. • Encrypted Indique si le volume est chiffré. • Type de système de fichiers Le type de système de fichiers Linux pour le volume. • IOPS Le nombre de I/O opération s par seconde (IOPS). • Clé KMS • Rôle IAM Le nom de l'ARN ou du rôle IAM associé à ce volume. • Size La taille du volume en Gio. • ID d'instantané ID d'instantané du volume. • Spécifications de balise La configuration des balises pour le volume. • Débit

Ressource	Description
	Le débit alloué pour le volume. • Type de volume Type de volume.
Service Connect	La configuration de Service Connect • Élément indiquant si Service Connect en cours d'utilisation • Espace de noms • Liste des services interconnectés • Configuration de la journalisation de Service Connect
Équilibreur de charge de service	Les équilibreurs de charge qui acheminent le trafic de service. • Nom • Nom du conteneur • Port du conteneur • ARN du groupe cible
Surveillance d'exécution	Indique si la surveillance d'exécution est activée.
Date de création	La date de création de la révision de service.

Ressource	Description
Treillis en PVC	La configuration du VPC Lattice pour la révision de service.
Contrôle	La configuration de surveillance du service, qui définit la résolution pour le niveau de service `CPUUtilization` et `MemoryUtilization` CloudWatch métriques. • Noms des métriques Les métriques à collecter (`CPUUtilization` , `MemoryUtilization`). • Résolution (secondes) Résolution de collecte des métriques. Les valeurs valides sont `20` et `60`.

Affichage des détails de la révision d'Amazon ECS

Vous pouvez consulter les informations relatives aux types de révisions de service suivants qui ont été créés le 25 octobre 2024 ou après cette date :

- Source : la configuration de la charge de travail actuellement déployée
- Cible : la configuration de la charge de travail en cours de déploiement

Amazon ECS Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

La page des détails du service s'affiche.

4. Sur la page des détails du service, choisissez Déploiements.
5. Choisissez la révision de service à consulter.

Pour consulter les révisions de service pour ce type de déploiement	Faites ceci
Déploiement en cours	Sous Déploiements en cours, procédez comme suit : • Pour consulter la révision de service source, sous Révisions de service, choisissez l'ID de révision de service pour le Type de révision source. • Pour consulter la révision de service cible, sous Révisions de service, choisissez l'ID de révision de service pour le Type de révision cible.
Dernier déploiement	Sous Dernier déploiement, choisissez Révision de service cible.
Déploiements terminés	Sous Déploiements de service, procédez comme suit : • Sous Révision du service cible, choisissez l'ID.

AWS CLI

1. Exécutez `describe-service-deployments` pour récupérer l'ARN de révision de service.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-deployments --service-deployment-arns
 arn:aws:ecs:region:account-id:service/cluster-name/service-name/NCWGC2ZR-
taawPAYrIaU5
```

   Notez l'`arn` des `sourceServiceRevisions` ou des `targetServiceRevisions`.

```
{
    "serviceDeployments": [
        {
            "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-
deployment/example/sd-example/NCWGC2ZR-taawPAYrIaU5",
            "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/example/
sd-example",
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/example",
            "updatedAt": "2024-09-10T16:49:35.572000+00:00",
            "sourceServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373578954",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "targetServiceRevision": {
                "arn": "arn:aws:ecs:us-west-2:123456789012:service-revision/
example/sd-example/4980306466373577095",
                "requestedTaskCount": 0,
                "runningTaskCount": 0,
                "pendingTaskCount": 0
            },
            "status": "IN_PROGRESS",
            "deploymentConfiguration": {
                "deploymentCircuitBreaker": {
                    "enable": false,
                    "rollback": false
                },
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
```

```
            }
        }
    ],
    "failures": []
}
```

2. Exécutez `describe-service-revisions`. Utilisez celui `arn` qui a été renvoyé par `describe-service-deployments`.

   Remplacez les variables par vos valeurs.

```
aws ecs describe-service-revisions --service-revision-arns
 arn:aws:ecs:region:123456789012:service-revision/cluster-name/service-
name/4980306466373577095
```

Équilibrage d’un service Amazon ECS entre les zones de disponibilité

À compter du 5 septembre 2025, Amazon ECS permet le rééquilibrage des zones de disponibilité pour tous les services éligibles à cette fonctionnalité. Un service est éligible lorsque la répartition entre les zones de disponibilité est la première stratégie de placement des tâches, ou lorsqu’il n’existe aucune stratégie de placement.

Pour aider vos applications à atteindre une haute disponibilité, nous vous recommandons de configurer vos services multitâches de manière à les exécuter dans plusieurs zones de disponibilité. Pour les services qui spécifient que leur première stratégie de placement est l'étalement des zones de disponibilité, AWS met tout en œuvre pour répartir équitablement les tâches de service entre les zones de disponibilité disponibles.

Toutefois, il peut arriver que le nombre de tâches exécutées dans une zone de disponibilité ne soit pas le même que dans les autres zones de disponibilité, par exemple après l’indisponibilité d’une zone de disponibilité. Pour corriger ce déséquilibre, vous pouvez activer la fonctionnalité de rééquilibrage des zones de disponibilité.

Grâce au rééquilibrage des zones de disponibilité, Amazon ECS surveille en permanence la répartition des tâches entre les zones de disponibilité pour chacun de vos services. Quand Amazon ECS détecte une répartition inégale des tâches, il prend automatiquement des mesures pour rééquilibrer la charge de travail entre les zones de disponibilité. Cela implique de lancer de nouvelles tâches dans les zones de disponibilité comportant le moins de tâches et de mettre fin à des tâches dans les zones de disponibilité surchargées.

Cette redistribution offre la garantie qu'aucune zone de disponibilité ne deviendra un point de défaillance, ce qui contribue à maintenir la disponibilité globale de vos applications conteneurisées. Le processus de rééquilibrage automatisé élimine la nécessité d'une intervention manuelle, réduisant ainsi le temps de reprise après un événement.

Voici un aperçu du processus de rééquilibrage des zones de disponibilité :

1. Amazon ECS commence à surveiller un service une fois qu'il a atteint l'état stable et examine le nombre de tâches exécutées dans chaque zone de disponibilité.
2. Amazon ECS effectue les opérations suivantes lorsqu'il détecte un déséquilibre dans le nombre de tâches exécutées dans chaque zone de disponibilité :
   - Il envoie un événement de service indiquant que le rééquilibrage de la zone de disponibilité commence.
   - Il lance les tâches dans les zones de disponibilité ayant le plus petit nombre de tâches en cours d'exécution.
   - Il arrête les tâches dans les zones de disponibilité ayant le plus grand nombre de tâches en cours d'exécution.
   - Le planificateur attend que les tâches nouvellement lancées soient `HEALTHY` et `RUNNING` avant d'arrêter les tâches dans la zone de disponibilité surdimensionnée.
   - Il envoie un événement de service avec le résultat du rééquilibrage de la zone de disponibilité.

Comment Amazon ECS détecte une répartition inégale des tâches

Amazon ECS détermine un déséquilibre dans le nombre de tâches exécutées dans chaque zone de disponibilité en divisant le nombre de tâches souhaité pour le service par le nombre de zones de disponibilité configurées. Si le nombre de tâches souhaité ne se répartit pas de manière égale, Amazon ECS répartit le reste des tâches de manière égale entre les zones de disponibilité configurées. Chaque zone de disponibilité doit avoir au moins une tâche.

Prenons l'exemple d'un service Amazon ECS avec un nombre souhaité de deux tâches configurées pour deux zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité est réparti de manière égale. Une distribution équilibrée serait une tâche par zone de disponibilité. S'il y a deux tâches dans la zone de disponibilité 1 et aucune tâche dans la zone de disponibilité 2, Amazon ECS initie le rééquilibrage en démarrant une tâche dans la zone de disponibilité 2 avant d'arrêter une tâche dans la zone de disponibilité 1.

À présent, imaginez un service Amazon ECS avec un nombre souhaité de trois tâches configurées pour deux zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité ne se divise pas de manière égale. Une distribution équilibrée comporterait une tâche dans la zone de disponibilité 1 et deux tâches dans la zone de disponibilité 2, car chaque zone de disponibilité comporte au moins une tâche et le reste de la tâche est placée dans la zone de disponibilité 2.

Prenons l'exemple d'un service Amazon ECS dont le nombre souhaité de cinq tâches est configuré pour trois zones de disponibilité. Dans ce scénario, le nombre de tâches souhaité ne se divise pas de manière égale. Une distribution équilibrée comporterait une tâche dans la zone de disponibilité 1 et deux tâches dans chacune des zones de disponibilité 2 et 3. Après avoir pris en compte chaque zone de disponibilité ayant une tâche chacune, les deux tâches restantes sont réparties de manière égale entre les zones de disponibilité.

Considérations relatives à la configuration du rééquilibrage des zones de disponibilité

Tenez compte des éléments suivants lorsque vous souhaitez configurer le rééquilibrage des zones de disponibilité :

- Le rééquilibrage des zones de disponibilité prend en charge les fournisseurs de capacité Fargate et EC2, et est pris en charge sur les instances gérées Amazon ECS. Pour Fargate, Amazon ECS redistribuera automatiquement les tâches entre les zones de disponibilité disponibles afin de maintenir l'équilibre. Pour les fournisseurs de capacité EC2, Amazon ECS rééquilibre au mieux les tâches entre les instances de conteneur existantes, en respectant les stratégies et les contraintes de placement que vous avez définies. Toutefois, Amazon ECS ne peut pas lancer de nouvelles instances dans les zones de disponibilité sous-utilisées dans le cadre du processus de rééquilibrage, ce qui limite le rééquilibrage aux instances de conteneur existantes.
- Le rééquilibrage des zones de disponibilité fonctionne dans les configurations suivantes :
  - Services utilisant la stratégie `Replica`
  - Services qui spécifient la répartition de la zone de disponibilité comme première stratégie de placement des tâches, ou qui ne spécifient pas de stratégie de placement.
- Vous ne pouvez pas utiliser le rééquilibrage des zones de disponibilité avec des services répondant à l'un des critères suivants :
  - Utilise la stratégie `Daemon`
  - Utilise le type de lancement `EXTERNAL` (ECS Anywhere)
  - Utilise 100 % pour la valeur `maximumPercent`
  - Utilise un Classic Load Balancer

- Utilise `attribute:ecs.availability-zone` comme contrainte de placement des tâches

Stratégies de placement et contraintes de placement avec rééquilibrage des zones de disponibilité

Les stratégies de placement déterminent la manière dont Amazon ECS sélectionne les instances de conteneur et les zones de disponibilité pour la fin du placement des tâches. Les contraintes de placement des tâches sont des règles qui déterminent si une tâche est autorisée à s'exécuter sur une instance de conteneur spécifique.

Pour EC2, vous pouvez utiliser des stratégies et des contraintes de placement en même temps que le rééquilibrage des zones de disponibilité. Cependant, pour que le rééquilibrage de la zone de disponibilité fonctionne, la stratégie de placement par répartition des zones de disponibilité doit être la première stratégie spécifiée.

Le rééquilibrage des zones de disponibilité est compatible avec différentes combinaisons de stratégies de placement. Par exemple, vous pouvez créer une stratégie qui distribue d'abord les tâches de manière uniforme entre les zones de disponibilité, puis regroupe les tâches par bin packing en fonction de la mémoire disponible dans chaque zone de disponibilité. Dans ce cas, le rééquilibrage de la répartition des zones de disponibilité fonctionne, car la stratégie de répartition des zones de disponibilité est spécifiée en premier.

Il est important de noter que le rééquilibrage de la zone de disponibilité ne fonctionnera pas si la première stratégie dans le tableau des stratégies de placement n'est pas du type « Répartition des zones de disponibilité ». Cette exigence garantit que l'objectif principal de la distribution des tâches est de maintenir l'équilibre entre les zones de disponibilité, ce qui est crucial pour une haute disponibilité.

Pour plus d'informations sur les stratégies et contraintes de placement des tâches, consultez la section Comment Amazon ECS place les tâches sur des instances de conteneur.

Les stratégies et contraintes de placement des tâches ne sont pas prises en charge pour les tâches utilisant Fargate. Fargate fera de son mieux pour répartir les tâches entre les zones de disponibilité accessibles. Si le fournisseur de capacité inclut à la fois Fargate et Fargate Spot, le comportement de répartition est indépendant pour chaque fournisseur de capacité.

La stratégie suivante répartit les tâches de manière uniforme entre les zones de disponibilité, puis regroupe les tâches par bin packing en fonction de la mémoire disponible dans chaque zone de

disponibilité. Le rééquilibrage des zones de disponibilité est compatible avec le service, car c'est la stratégie `spread` qui prime.

```
"placementStrategy": [
    {
        "field": "attribute:ecs.availability-zone",
        "type": "spread"
    },
    {
        "field": "memory",
        "type": "binpack"
    }
]
```

Activation du rééquilibrage des zones de disponibilité

Vous devez activer le rééquilibrage des zones de disponibilité pour les services nouveaux et existants.

Vous pouvez activer et désactiver le rééquilibrage de la zone de disponibilité à l'aide de la console, des API, AWS CLI, et CloudFormation.

Le comportement par défaut de `AvailabilityZoneRebalancing` diffère entre les requêtes de création et de mise à jour :

- Pour créer des requêtes de service, lorsqu'aucune valeur n'est spécifiée pour `AvailabilityZoneRebalancing`, Amazon ECS définit la valeur par défaut sur `ENABLED`.
- Pour les requêtes de service de mise à jour, lorsqu'aucune valeur n'est spécifiée pour `AvailabilityZoneRebalancing`, Amazon ECS utilise par défaut la valeur de `AvailabilityZoneRebalancing` du service existant. Si le service n'a jamais défini de valeur pour `AvailabilityZoneRebalancing`, Amazon ECS la traite comme `DISABLED`.

Type de service	API	Console	INTERFACE DE LIGNE DE COMMANDE (CLI)
Existing	UpdateService	Mettre à jour un service Amazon ECS	update-service

Type de service	API	Console	INTERFACE DE LIGNE DE COMMANDE (CLI)
Nouveau	CreateService	Création d'un déploiement de mise à jour propagée Amazon ECS	create-service

L'exemple suivant montre comment activer le rééquilibrage des services lors de la création d'un service :

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-definition:1 \
    --desired-count 6 \
    --availability-zone-rebalancing ENABLED
```

Suivi du rééquilibrage des zones de disponibilité Amazon ECS

Vous pouvez vérifier si le rééquilibrage des zones de disponibilité est activé pour un service dans la console ou en appelant `describe-services`. L'exemple suivant peut être utilisé pour consulter le statut à l'aide de la CLI.

La réponse sera soit `ENABLED`, soit `DISABLED`.

```
aws ecs describe-services \
    --services service-name \
    --cluster cluster-name \
    --query services[0].availabilityZoneRebalancing
```

Événements de service

Amazon ECS envoie des événements d'action de service pour vous aider à comprendre le cycle de vie du rééquilibrage des zones de disponibilité.

Événement	Scénario	Type	En savoir plus
SERVICE_REBALANCING_STARTED	Amazon ECS lance une opération de rééquilibrage des zones de disponibilité	INFO	service (*service-name*) n'est pas équilibré en Z avec *tâches numériques* tâches dans *Zone de disponibilité 1* , *tâches numériques* in *Zone de disponibilité 2* , et *tâches numériques* in *Zone de disponibilité 3* . AZ Rééquilibrage en cours.
SERVICE_REBALANCING_COMPLETED	L'opération de rééquilibrage des zones de disponibilité est terminée	INFO	service (*service-name*) est équilibré en A avec *tâches numériques* tâches dans *Zone de disponibilité 1* , *tâches numériques* tâches dans *Zone de disponibilité 2* , et *tâches numériques* tâches dans *Zone de disponibilité 3* .
TASKS_STARTED	Amazon ECS démarre avec succès les tâches dans le cadre de l'opération	INFO	*service-name* a commencé *tâches numériques* tâches

Événement	Scénario	Type	En savoir plus
	de rééquilibrage de la zone de disponibilité		dans *Zone de disponibilité* à AZ Rebalance : *identifiants de tâches*.
TASKS_STOPPED	Amazon ECS arrête avec succès les tâches dans le cadre de l'opération de rééquilibrage de la zone de disponibilité	INFO	*service-name* s'est arrêté *tâches numériques* exécution de tâches dans *Zone de disponibilité* grâce au rééquilibrage AZ : *identifiant de tâche*.
SERVICE_TASK_PLACEMENT_FAILURE	Amazon ECS n'a pas pu démarrer une tâche dans le cadre de l'opération de rééquilibrage de la zone de disponibilité	ERROR	Pour EC2, consultez la section service (*service-name*) ne parvient pas à placer une tâche dans *Zone de disponibilité* car aucune instance de conteneur ne répondait à toutes ses exigences. Pour Fargate, consultez la section service (*service-name*) ne parvient pas à placer une tâche dans *Zone de disponibilité*.

Événement	Scénario	Type	En savoir plus
TASKSET_SCALE_IN_FAILURE_BY_TASK_PROTECTION	L'opération de rééquilibrage de la zone de disponibilité est bloquée, car la protection des tâches est utilisée.	INFO	service (*service-name*) n'a pas pu effectuer AZ Rebalance car *nom de l'ensemble de tâches* n'a pas pu évoluer en raison de *raison*.
SERVICE_REBALANCING_STOPPED	L'opération de rééquilibrage des zones de disponibilité s'est arrêtée. Amazon ECS envoie des événements supplémentaires qui fournissent davantage d'informations.	INFO	service (*service-name*) a arrêté AZ Rebalancing.

Événements de modification de l'état de la tâche

Amazon ECS envoie un événement de changement d'état de tâche (`START`) pour chaque tâche qu'il démarre dans le cadre du processus de rééquilibrage.

Amazon ECS envoie un événement de changement d'état de tâche (`STOPPED`) pour chaque tâche qu'il arrête dans le cadre du processus de rééquilibrage. La raison est définie sur `Availability-zone rebalancing initiated by (deployment ecs-svc/`*`deployment-id`*`)`.

Pour plus d'informations sur les événements, consultez la section Événements de changement d'état d'une tâche Amazon ECS.

Résolution des problèmes de rééquilibrage des services

Si vous rencontrez des problèmes lors du rééquilibrage des services, envisagez les étapes de résolution des problèmes suivantes :

Le rééquilibrage ne démarre pas

Vérifiez que :

- le rééquilibrage des services est activé pour votre service ;
- votre service utilise une configuration prise en charge (consultez la section the section called “Considérations relatives à la configuration du rééquilibrage des zones de disponibilité”) ;
- votre service a atteint un état stable.

Échecs de placement des tâches pendant le rééquilibrage

Si vous voyez des événements `SERVICE_TASK_PLACEMENT_FAILURE` :

- Pour EC2 : vérifiez si des instances de conteneur sont disponibles dans la zone de disponibilité cible.
- Pour Fargate : vérifiez si des contraintes de ressources ou des quotas de service limitent le placement des tâches.
- Passez en revue vos contraintes de placement des tâches pour vous assurer qu’elles n’empêchent pas une bonne répartition des tâches.

Le rééquilibrage s’arrête de façon inattendue

Si vous voyez des événements `SERVICE_REBALANCING_STOPPED` :

- Vérifiez s’il existe une protection des tâches susceptible de bloquer l’opération.
- Recherchez les déploiements de service simultanés susceptibles d’interrompre le rééquilibrage.
- Consultez les événements de service pour obtenir des informations supplémentaires sur les raisons pour lesquelles le rééquilibrage a été arrêté.

Pratiques exemplaires en matière de rééquilibrage des services

Suivez ces pratiques exemplaires pour tirer le meilleur parti du rééquilibrage des services :

- Surveiller les opérations de rééquilibrage - Configurer CloudWatch des alarmes pour surveiller les événements de service liés au rééquilibrage afin d'identifier rapidement tout problème.
- Considérer l’impact sur les performances : sachez que les opérations de rééquilibrage peuvent augmenter temporairement l’utilisation des ressources lorsque de nouvelles tâches sont lancées avant que les anciennes ne soient arrêtées.
- Utiliser la protection des tâches de manière stratégique : si vous avez des tâches critiques qui ne doivent pas être résiliées pendant le rééquilibrage, envisagez d’utiliser la protection des tâches.

- Planifier la capacité EC2 : pour EC2, assurez-vous de disposer de suffisamment d'instances de conteneurs dans toutes les zones de disponibilité pour permettre un rééquilibrage efficace.
- Tester le comportement du rééquilibrage : avant de vous fier au rééquilibrage en production, testez le comportement de vos services lors des opérations de rééquilibrage dans un environnement hors production.

Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS

En option, votre service peut être configuré pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches de votre service.

> **Note**
>
> Lorsque vous utilisez des ensembles de tâches, toutes les tâches de l'ensemble doivent être configurées pour utiliser Elastic Load Balancing ou pour ne pas utiliser Elastic Load Balancing.

Services Amazon ECS hébergés sur AWS Fargate prennent en charge les équilibreurs de charge d'application, les équilibreurs de charge réseau et les équilibreurs de charge de passerelle. Utilisez le tableau suivant pour savoir quel type d'équilibreur de charge utiliser.

Type d'équilibreur de charge	À utiliser dans les cas suivants	
Application Load Balancer	Parcours HTTP/HTTPS trafic (ou couche 7). Les équilibreurs de charge Application Load Balancer offrent plusieurs fonctions qui les rendent intéressants à utiliser avec les services Amazon ECS service : • Chaque service peut desservir le trafic provenant	

Type d’équilibreur de charge	À utiliser dans les cas suivants
	de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles. • Ils sont pris en charge par des tâches hébergées sur des instances Fargate ou EC2. • Les Application Load Balancers permettent aux conteneurs d'utiliser le mappage de port hôte dynamique (de sorte que plusieurs tâches du même service soient autorisées par instance de conteneur). • Les Application Load Balancers prennent en charge le routage basé sur le chemin d'accès et les règles de priorité (pour que plusieurs services puissent utiliser le même port d'écoute sur un même Application Load Balancer).
Network Load Balancer	Acheminer le trafic TCP ou UDP (ou couche 4).

Type d’équilibreur de charge	À utiliser dans les cas suivants
Passerelle équilibreur de charge	Acheminer le trafic TCP ou UDP (ou couche 4). Utiliser des appareils virtuels, tels que des pare-feu, des systèmes de détection et de prévention des intrusions et des systèmes d’inspection approfondie des paquets.

Nous vous recommandons d’utiliser des Application Load Balancer pour vos services Amazon ECS afin de pouvoir profiter des fonctions les plus récentes, sauf si votre service nécessite une fonctionnalité qui n’est fournie que par les Network Load Balancer ou les Gateway Load Balancer. Pour plus d'informations sur Elastic Load Balancing et les différences entre les types d'équilibreurs de charge, consultez le Guide de l'utilisateur Elastic Load Balancing.

Avec votre équilibreur de charge, vous payez uniquement en fonction de votre utilisation. Pour plus d'informations, veuillez consultez Tarification Elastic Load Balancing.

Optimisation des paramètres de surveillance de l’état de l’équilibreur de charge pour Amazon ECS

Les équilibreurs de charge n’acheminent les requêtes que vers les cibles saines dans les zones de disponibilité de l’équilibreur de charge. Chaque cible est enregistrée dans un groupe cible. Chaque nœud d’équilibreur de charge vérifie l’état de chaque cible en utilisant les paramètres de surveillance de l’état de chaque groupe cible auprès duquel la cible est enregistrée. Une fois que votre cible est enregistrée, elle doit passer avec succès une seule surveillance de l’état pour être considérée comme saine. Amazon ECS surveille l’équilibreur de charge. L’équilibreur de charge envoie régulièrement des surveillances de l’état au conteneur Amazon ECS. L’agent Amazon ECS surveille et attend que l’équilibreur de charge publie un rapport sur l’état du conteneur. Il le fait avant de considérer que le conteneur est sain.

Deux paramètres de surveillance de l’état Elastic Load Balancing affectent la vitesse de déploiement :

- Intervalle entre les surveillances de l'état : détermine la durée approximative, en secondes, entre les surveillances de l'état d'un conteneur individuel. Par défaut, l'équilibreur de charge vérifie toutes les 30 secondes.

  Ce paramètre est nommé :

  - `HealthCheckIntervalSeconds` dans l'API Elastic Load Balancing
  - Intervalle sur la console Amazon EC2
- Seuil de comptage de l'état sain : détermine le nombre de surveillances de l'état consécutives réussies requises avant de considérer un conteneur défectueux comme sain. Par défaut, l'équilibreur de charge nécessite cinq surveillances de l'état réussies avant de signaler que le conteneur cible est sain.

  Ce paramètre est nommé :

  - `HealthyThresholdCount` dans l'API Elastic Load Balancing
  - Seuil de l'état sain sur la console Amazon EC2

Important : pour les cibles nouvellement enregistrées, une seule surveillance de l'état réussie suffit pour considérer que la cible est saine, quel que soit le seuil de l'état sain défini. Le seuil de comptage de l'état sain ne s'applique que lorsqu'une cible passe d'un état défectueux à un état sain.

Avec les paramètres par défaut, si une cible devient défectueuse puis se rétablit, le temps total nécessaire pour déterminer l'état d'un conteneur est de 2 minutes et 30 secondes (`30 seconds * 5 = 150 seconds`).

Vous pouvez accélérer le processus de surveillance de l'état si votre service démarre et se stabilise en moins de 10 secondes. Pour accélérer le processus, réduisez l'intervalle entre les surveillances de l'état et le seuil de comptage de l'état sain.

- `HealthCheckIntervalSeconds` (nom pour l'API Elastic Load Balancing) ou Interval (nom pour la console Amazon EC2) : 5
- `HealthyThresholdCount` (nom pour l'API Elastic Load Balancing) ou seuil sain (nom pour la console Amazon EC2) : 2

Avec ce paramètre, le processus de vérification de l'état prend 10 secondes, contre 2 minutes et 30 secondes par défaut.

Pour plus d'informations sur les paramètres de surveillance de l'état Elastic Load Balancing, consultez la section Surveillance de l'état de vos groupes cibles dans le Guide de l'utilisateur Elastic Load Balancing.

Optimisation des paramètres de drainage de la connexion de l'équilibreur de charge pour Amazon ECS

Pour permettre l'optimisation, les clients maintiennent une connexion persistante au service de conteneurs. Cela permet aux requêtes ultérieures de ce client de réutiliser la connexion existante. Lorsque vous souhaitez arrêter le trafic vers un conteneur, vous devez en informer l'équilibreur de charge. L'équilibreur de charge vérifie régulièrement si le client a fermé la connexion persistante. Amazon ECS surveille l'équilibreur de charge et attend qu'il indique que la connexion persistante est fermée (la cible est dans un état `UNUSED`).

La durée pendant laquelle l'équilibreur de charge attend pour faire passer la cible à l'état `UNUSED` correspond au délai d'annulation de l'enregistrement. Vous pouvez configurer le paramètre d'équilibreur de charge suivant pour accélérer vos déploiements.

- `deregistration_delay.timeout_seconds` : 300 (par défaut)

Lorsque vous disposez d'un service dont le temps de réponse est inférieur à une seconde, définissez le paramètre sur la valeur suivante pour que l'équilibreur de charge n'attende que cinq secondes avant de rompre la connexion entre le client et le service principal :

- `deregistration_delay.timeout_seconds` : 5

> **Note**
>
> Ne définissez pas la valeur sur cinq secondes lorsqu'un service comporte des requêtes de longue durée, telles que des téléchargements de fichiers lents ou des connexions de streaming.

Réactivité à SIGTERM

Amazon ECS envoie d'abord un signal d'arrêt à la tâche pour indiquer que l'application doit se terminer et s'arrêter. Ce signal peut être défini dans l'image de votre conteneur à l'aide de l'instruction STOPSIGNAL et sera par défaut SIGTERM. Amazon ECS envoie ensuite un message SIGKILL.

Lorsque les applications ignorent le signal SIGTERM, le service Amazon ECS doit attendre pour envoyer le signal SIGKILL afin de terminer le processus.

La durée pendant laquelle Amazon ECS attend avant d'envoyer le message SIGKILL est déterminée par l'option d'agent Amazon ECS suivante :

- `ECS_CONTAINER_STOP_TIMEOUT` : 30 (par défaut)

  Pour plus d'informations sur le paramètre de l'agent de conteneur, consultez Agent de conteneurs Amazon ECS sur GitHub.

Pour accélérer le délai d'attente, définissez le paramètre de l'agent Amazon ECS sur la valeur suivante :

- `ECS_CONTAINER_STOP_TIMEOUT` : 2

  Si votre application prend plus d'une seconde, multipliez la valeur par deux et utilisez ce nombre comme valeur.

Dans ce cas, Amazon ECS attend deux secondes que le conteneur s'arrête, puis Amazon ECS envoie un message SIGKILL lorsque l'application ne s'arrête pas.

Vous pouvez également modifier le code de l'application pour intercepter le signal SIGTERM et y réagir. Ce qui suit est un exemple dans JavaScript:

```
process.on('SIGTERM', function() {
  server.close();
})
```

Ce code fait en sorte que le serveur HTTP cesse d'écouter les nouvelles demandes, arrête de répondre à toutes les demandes en cours de traitement, puis Node.js le processus se termine car la boucle d'événements n'a rien à voir. Cela étant, si le processus ne prend que 500 ms pour terminer ses requêtes en cours de vol, il se termine prématurément sans avoir à attendre la fin du délai d'arrêt et sans recevoir de SIGKILL.

Utilisation d'un Application Load Balancer pour Amazon ECS

Un équilibreur de charge d'application prend des décisions de routage au niveau de la couche application (HTTP/HTTPS), prend en charge le routage basé sur les chemins et peut acheminer

les demandes vers un ou plusieurs ports de chaque instance de conteneur de votre cluster. Les équilibreurs de charge Application Load Balancer prennent en charge le mappage de port hôte dynamique. Par exemple, si la définition de conteneur de votre tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement dans la plage de ports éphémères de l'instance de conteneur (par exemple 32768 à 61000 sur la dernière version d'Amazon). ECS-optimized AMI). Au lancement de la tâche, le conteneur NGINX est enregistré auprès de l'Application Load Balancer sous la forme d'une combinaison d'ID d'instance et de port, et le trafic est distribué vers l'ID d'instance et le port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, consultez le Guide de l'utilisateur pour les équilibreurs de charge Application Load Balancer.

Pour plus d'informations sur les pratiques exemplaires en matière de définition de paramètres pour accélérer vos déploiements, consultez la section :

- Optimisation des paramètres de surveillance de l'état de l'équilibreur de charge pour Amazon ECS
- Optimisation des paramètres de drainage de la connexion de l'équilibreur de charge pour Amazon ECS

Tenez compte des éléments suivants lorsque vous utilisez des Application Load Balancer avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- Pour les services dans un IPv6-only configuration, vous devez définir le type d'adresse IP du groupe cible de l'équilibreur de charge d'application sur `dualstack` ou `dualstack-without-public-ipv4`.
- Pour les services avec des tâches utilisant le mode réseau `awsvpc`, lorsque vous créez un groupe cible pour votre service, vous devez choisir `ip` comme type de cible, pas `instance`. Cela est dû au fait que les tâches qui utilisent le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
- Si votre service nécessite l'accès à plusieurs ports à charge équilibrée, tels que le port 80 et le port 443 pour HTTP/HTTPS service, vous pouvez configurer deux écouteurs. Un écouteur se charge du protocole HTTPS qui transmet la demande au service et un autre écouteur est responsable

de la redirection des demandes HTTP vers le port HTTPS approprié. Pour plus d'informations, consultez la section Création d'un écouteur pour votre Application Load Balancer dans le Guide de l'utilisateur pour les Application Load Balancers.

- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la Console de gestion AWS. Vous pouvez utiliser le AWS Copilote, AWS CloudFormation, AWS CLI ou SDK pour modifier la configuration de l'équilibreur de charge pour `ECS` contrôleur de déploiement roulant uniquement, non AWS CodeDeploy blue/green ou externe. Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la façon de modifier la configuration, voir UpdateService dans le Référence de l'API Amazon Elastic Container Service.
- Si une tâche d’un service ne satisfait pas aux critères de surveillance de l’état de l’équilibreur de charge, la tâche est arrêtée et redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS.
- Lorsque vous utilisez le type de cible `instance`, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l’utilisation du type de cible `ip`, la connectivité entre VPC est prise en charge.
- Utilisez un groupe cible unique pour chaque service.

  L’utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de service.
- Vous devez spécifier les groupes cibles associés à un équilibreur de charge d'application.

Pour plus d’informations sur la création d’un Application Load Balancer, consultez la section Création d’un Application Load Balancer dans Application Load Balancer.

Utilisation d’un Network Load Balancer pour Amazon ECS

Un Network Load Balancer prend des décisions de routage au niveau de la couche transport (TCP/SSL). Il est capable de traiter des millions de demandes par seconde. Lorsque l'équilibreur de charge reçoit une connexion, il sélectionne une cible depuis le groupe cible pour la règle par défaut à l'aide d'un algorithme de routage du hachage de flux. Il tente d'ouvrir une connexion TCP à la cible sélectionnée sur le port spécifié dans la configuration de l'écouteur. Il transmet la demande sans modifier les en-têtes. Les équilibreurs de charge Network Load Balancer prennent en charge le mappage de port hôte dynamique. Par exemple, si la définition de conteneur de votre tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement dans la plage de ports éphémères de l'instance de conteneur (par exemple 32768 à 61000 sur la dernière version d'Amazon). ECS-optimized AMI). Lorsque la tâche est lancée, le conteneur NGINX est enregistré dans l'équilibreur de charge Network Load Balancer en tant que combinaison d'ID d'instance et de port, et le trafic est réparti vers l'ID d'instance et le port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, veuillez consulter le Guide de l'utilisateur pour les Network Load Balancers.

Pour plus d’informations sur les pratiques exemplaires en matière de définition de paramètres pour accélérer vos déploiements, consultez la section :

- Optimisation des paramètres de surveillance de l’état de l’équilibreur de charge pour Amazon ECS
- Optimisation des paramètres de drainage de la connexion de l’équilibreur de charge pour Amazon ECS

Tenez compte des informations suivantes lors de l’utilisation d’un Network Load Balancer avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour plus d’informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- Vous ne pouvez pas attacher plus de cinq groupes cibles à un service.
- Pour les services dans un IPv6-only configuration, vous devez définir le type d'adresse IP du groupe cible du Network Load Balancer sur `dualstack`.
- Pour les services avec des tâches utilisant le mode réseau `awsvpc`, lorsque vous créez un groupe cible pour votre service, vous devez choisir `ip` comme type de cible, pas `instance`. Cela est dû

au fait que les tâches qui utilisent le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.

- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la Console de gestion AWS. Vous pouvez utiliser le AWS Copilote, AWS CloudFormation, AWS CLI ou SDK pour modifier la configuration de l'équilibreur de charge pour `ECS` contrôleur de déploiement roulant uniquement, non AWS CodeDeploy blue/green ou externe. Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la façon de modifier la configuration, voir UpdateService dans le Référence de l'API Amazon Elastic Container Service.
- Si une tâche d'un service ne satisfait pas aux critères de surveillance de l'état de l'équilibreur de charge, la tâche est arrêtée et redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Lorsque vous utilisez un Gateway Load Balancer configuré avec des adresses IP comme cibles et la préservation de l'adresse IP du client désactivée, les requêtes sont considérées comme provenant de l'adresse IP privée du Gateway Load Balancer. Cela signifie que les services derrière un Gateway Load Balancer sont effectivement ouverts au monde dès que vous autorisez les requêtes entrantes et les surveillances de l'état dans le groupe de sécurité de la cible.
- Pour les tâches Fargate, vous devez utiliser la version de plateforme `1.4.0` (Linux) ou `1.0.0` (Windows).
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS.
- Lorsque vous utilisez le type de cible `instance`, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l'utilisation du type de cible `ip`, la connectivité entre VPC est prise en charge.
- La préservation des adresses IP client du Network Load Balancer est compatible avec les cibles Fargate.
- Utilisez un groupe cible unique pour chaque service.

L'utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de service.

- Vous devez spécifier les groupes cibles associés à un équilibreur de charge réseau.

Pour plus d'informations sur la création d'un Network Load Balancer, consultez la section Création d'un Network Load Balancer dans Network Load Balancer.

> **Important**
>
> Si la définition de tâche de votre service utilise le mode réseau `awsvpc` (ce qui est requis pour Fargate), vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
> Vous ne pouvez pas enregistrer des instances par ID d'instance si elles ont les types d'instance suivants : C1, CC1, CC2, CG1, GC2, CR1, G1, G2, HI1 et HS1, M1, M2, M3 et T1. Vous pouvez enregistrer les instances de ces types par adresse IP.

Utilisation d'un Gateway Load Balancer pour Amazon ECS

Les équilibreurs Gateway Load Balancer agissent au niveau de la troisième couche du modèle OSI Open Systems Interconnection (OSI), la couche réseau. Ils écoutent tous les paquets IP sur tous les ports et transfèrent le trafic vers le groupe cible spécifié dans la règle d'écoute. Il maintient l'adhérence des flux à une appliance cible spécifique en utilisant 5 tuples (pour TCP/UDP flux) ou 3 tuples (pour les non-TCP/UDP flux). Par exemple, si la définition de conteneur de votre tâche spécifie le port 80 pour un port de conteneur NGINX et le port 0 pour le port hôte, le port hôte est choisi dynamiquement dans la plage de ports éphémères de l'instance de conteneur (par exemple 32768 à 61000 sur la dernière version d'Amazon). ECS-optimized AMI). Au lancement de la tâche, le conteneur NGINX est enregistré auprès du Gateway Load Balancer sous la forme d'une combinaison d'ID d'instance et de port, et le trafic est distribué vers l'ID d'instance et le port correspondant à ce conteneur. Ce mappage dynamique vous permet d'exécuter plusieurs tâches à partir d'un seul service sur la même instance de conteneur. Pour plus d'informations, consultez la section Qu'est-ce qu'un Gateway Load Balancer dans Gateway Load Balancer.

Pour plus d'informations sur les pratiques exemplaires en matière de définition de paramètres pour accélérer vos déploiements, consultez la section :

- Optimisation des paramètres de surveillance de l'état de l'équilibreur de charge pour Amazon ECS
- Optimisation des paramètres de drainage de la connexion de l'équilibreur de charge pour Amazon ECS

Tenez compte des informations suivantes lors de l'utilisation de Gateway Load Balancer avec Amazon ECS :

- Amazon ECS nécessite le rôle IAM lié au service qui fournit les autorisations nécessaires pour enregistrer et désenregistrer des cibles dans votre équilibreur de charge lors de la création ou de l'arrêt de tâches. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.
- Pour les services dans un IPv6-only configuration, vous devez définir le type d'adresse IP du groupe cible du Gateway Load Balancer sur `dualstack`.
- Pour les services dont les tâches utilisent un mode réseau autre que `awsvpc`, les Gateway Load Balancer ne sont pas pris en charge.
- La configuration de sous-réseau de votre équilibreur de charge doit inclure toutes les zones de disponibilité dans lesquels résident vos instances de conteneur.
- Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la Console de gestion AWS. Vous pouvez utiliser le AWS Copilote, AWS CloudFormation, AWS CLI ou SDK pour modifier la configuration de l'équilibreur de charge pour `ECS` contrôleur de déploiement roulant uniquement, non AWS CodeDeploy blue/green ou externe. Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la façon de modifier la configuration, voir UpdateService dans le Référence de l'API Amazon Elastic Container Service.
- Si une tâche d'un service ne satisfait pas aux critères de surveillance de l'état de l'équilibreur de charge, la tâche est arrêtée et redémarrée. Ce processus continue jusqu'à ce que votre service atteigne le nombre souhaité de tâches en cours d'exécution.
- Lorsque vous utilisez un Gateway Load Balancer configuré avec des adresses IP comme cibles, les requêtes sont considérées comme provenant de l'adresse IP privée du Gateway Load Balancer. Cela signifie que les services derrière un Gateway Load Balancer sont effectivement

ouverts au monde dès que vous autorisez les requêtes entrantes et les surveillances de l'état dans le groupe de sécurité de la cible.

- Pour les tâches Fargate, vous devez utiliser la version de plateforme `1.4.0` (Linux) ou `1.0.0` (Windows).
- Si vous rencontrez des problèmes avec l'équilibreur de charge utilisé par vos services, consultez Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS.
- Lorsque vous utilisez le type de cible `instance`, vos tâches et votre équilibreur de charge doivent se trouver dans le même VPC. Lors de l'utilisation du type de cible `ip`, la connectivité entre VPC est prise en charge.
- Utilisez un groupe cible unique pour chaque service.

  L'utilisation du même groupe cible pour plusieurs services peut entraîner des problèmes lors des déploiements de service.
- Vous devez spécifier les groupes cibles associés à un Gateway Load Balancer.

Pour plus d'informations sur la création d'un Gateway Load Balancer, consultez la section Démarrage avec les Gateway Load Balancer dans Gateway Load Balancer.

> **Important**
>
> Si la définition de tâche de votre service utilise le mode réseau `awsvpc` (ce qui est requis pour Fargate), vous devez choisir le type de cible `ip`, et non `instance`. Cela est dû au fait que les tâches qui utilisent le mode réseau `awsvpc` sont associées à une interface réseau Elastic et non à une instance Amazon EC2.
> Vous ne pouvez pas enregistrer des instances par ID d'instance si elles ont les types d'instance suivants : C1, CC1, CC2, CG1, GC2, CR1, G1, G2, HI1 et HS1, M1, M2, M3 et T1. Vous pouvez enregistrer les instances de ces types par adresse IP.

Enregistrement de plusieurs groupes cibles auprès d'un service Amazon ECS

Votre service Amazon ECS service peut desservir le trafic provenant de plusieurs équilibreurs de charge et exposer plusieurs ports à charge équilibrée en spécifiant plusieurs groupes cibles dans une définition de service.

Pour créer un service spécifiant plusieurs groupes cibles, vous devez le créer à l'aide de l'API Amazon ECS, du SDK, AWS CLI, ou un CloudFormation modèle. Une fois que le service est créé,

vous pouvez afficher le service et les groupes cibles enregistrés auprès de celui-ci avec la Console de gestion AWS. Vous devez utiliser `UpdateService` pour modifier la configuration de l'équilibreur de charge d'un service existant.

Plusieurs groupes cibles peuvent être spécifiés dans une définition de service à l'aide du format suivant. Pour obtenir la syntaxe complète d'une définition de service, consultez Modèle de définition de service.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"container_name",
      "containerPort":container_port
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"container_name",
      "containerPort":container_port
   }
]
```

Considérations

Les informations suivantes doivent être prises en compte lorsque vous spécifiez plusieurs groupes cibles dans une définition de service.

- Pour les services qui utilisent un équilibreur Application Load Balancer ou un Network Load Balancer, vous ne pouvez pas attacher plus de cinq groupes cibles à un service.
- La spécification de plusieurs groupes cibles dans une définition de service n'est prise en charge que dans les conditions suivantes :
  - Le service doit utiliser un équilibreur de charge Application Load Balancer ou un Network Load Balancer.
  - Le service doit utiliser le type de contrôleur de déploiement (ECS). Il peut s'agir de l'Amazon ECS native/blue déploiement écologique, ou déploiement de mises à jour continues.
- La spécification de groupes cibles multiples est prise en charge pour des services contenant des tâches qui utilisent à la fois les types de lancement Fargate et EC2.

- Lors de la création d'un service qui spécifie plusieurs groupes cibles, le rôle lié à un service Amazon ECS service doit être créé. Le rôle est créé en omettant le paramètre `role` dans les demandes d'API, ou la propriété `Role` dans CloudFormation. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.

Exemples de définitions de service

Voici quelques exemples de cas d'utilisation pour spécifier plusieurs groupes cibles dans une définition de service. Pour obtenir la syntaxe complète d'une définition de service, consultez Modèle de définition de service.

Utilisation d'équilibreurs de charge distincts pour le trafic interne et externe

Dans le cas d'utilisation suivant, un service utilise deux équilibreurs de charge distincts, l'un pour le trafic interne et l'autre pour le trafic Internet, pour le même conteneur et port.

```
"loadBalancers":[
   //Internal ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"nginx",
      "containerPort":8080
   },
   //Internet-facing ELB
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"nginx",
      "containerPort":8080
   }
]
```

Exposition de plusieurs ports à partir du même conteneur

Dans le cas d'utilisation suivant, un service utilise un équilibreur de charge, mais expose plusieurs ports à partir du même conteneur. Par exemple, un conteneur Jenkins expose le port 8080 pour l'interface web Jenkins et le port 50000 pour l'API.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"jenkins",
      "containerPort":8080
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"jenkins",
      "containerPort":50000
   }
]
```

Exposition de ports à partir de plusieurs conteneurs

Dans le cas d'utilisation suivant, un service utilise un équilibreur de charge et deux groupes cibles pour exposer les ports de conteneurs distincts.

```
"loadBalancers":[
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_1/1234567890123456",
      "containerName":"webserver",
      "containerPort":80
   },
   {

 "targetGroupArn":"arn:aws:elasticloadbalancing:region:123456789012:targetgroup/
target_group_name_2/6543210987654321",
      "containerName":"database",
      "containerPort":3306
   }
]
```

Mise à l’échelle automatique de votre service Amazon ECS

La mise à l’échelle automatique est l’aptitude à augmenter ou à diminuer automatiquement le nombre souhaité de tâches dans votre service Amazon ECS. Amazon ECS utilise le service Application Auto Scaling pour fournir cette fonctionnalité. Pour de plus amples informations, veuillez consulter le Guide de l'utilisateur Application Auto Scaling.

Amazon ECS publie CloudWatch des statistiques relatives à l'utilisation moyenne du processeur et de la mémoire de votre service. Pour plus d’informations, consultez Métriques d’utilisation d’un service Amazon ECS. Vous pouvez les utiliser et d'autres CloudWatch des indicateurs permettant d'étendre votre service (ajouter des tâches supplémentaires) afin de faire face à une forte demande aux heures de pointe, et de faire évoluer votre service (exécuter moins de tâches) afin de réduire les coûts pendant les périodes de faible utilisation.

Amazon ECS Service Auto Scaling prend en charge les types de scalabilité automatique suivants :

- Utilisation d’une métrique cible pour mettre à l’échelle les services Amazon ECS – Augmente ou réduit le nombre de tâches exécutées par votre service en fonction d'une valeur cible pour une métrique spécifique. Cette option est similaire à la façon dont votre thermostat maintient la température de votre domicile. Vous sélectionnez une température et le thermostat se charge du reste.
- Utilisez des incréments prédéfinis en fonction de CloudWatch alarmes pour faire évoluer les services Amazon ECS – Augmente ou réduit le nombre de tâches exécutées par votre service en fonction d'un ensemble d'ajustements de mise à l'échelle, appelés ajustements d'étape, qui varient en fonction de la valeur du seuil de l'alarme.
- Utilisation des actions planifiées pour mettre à l’échelle les services Amazon ECS : augmente ou réduit le nombre de tâches exécutées par votre service en fonction de la date et de l’heure.
- Utilisation des tendances historiques pour mettre à l’échelle les services Amazon ECS via une mise à l’échelle prédictive : augmente ou réduit le nombre de tâches exécutées par votre service en fonction de l’analytique des données historiques de charge afin de détecter les tendances quotidiennes ou hebdomadaires dans les flux de trafic.

- Faites évoluer les services Amazon ECS en fonction d'une file d'attente Amazon SQS—Augmentez ou diminuez le nombre de tâches exécutées par votre service en fonction de l'arriéré de messages dans une file d'attente Amazon SQS, à l'aide d'une métrique personnalisée de backlog par tâche. Cela est utile pour les applications asynchrones dans lesquelles les tâches Amazon ECS agissent comme des travailleurs qui consomment les messages d'une file d'attente.

Considérations

Lorsque vous utilisez des stratégies de mise à l'échelle, tenez compte des informations suivantes :

- Amazon ECS envoie des métriques à intervalles d'une minute à CloudWatch. Les métriques ne sont pas disponibles tant que les clusters et les services ne les envoient pas à CloudWatch, et vous ne pouvez pas créer CloudWatch alarmes pour les métriques qui n'existent pas.
- Les stratégies de mise à l'échelle prennent en charge un temps de stabilisation. Il s'agit de la durée, en secondes, à attendre qu'une activité de mise à l'échelle précédente prenne effet.
  - Pour les événements de montée en puissance, l'intention est de réduire continuellement (mais pas excessivement) la montée en puissance. Une fois que Service Auto Scaling a réussi une montée en puissance à l'aide d'une stratégie de mise à l'échelle, l'application commence à calculer le temps de stabilisation. La politique de mise à l'échelle n'augmente pas à nouveau la capacité souhaitée, sauf si une plus grande montée en puissance est lancée ou si le temps de stabilisation est écoulé. Tandis que le temps de stabilisation de la montée en puissance s'applique, la capacité ajoutée par l'activité de mise à l'échelle initiale est calculée dans le cadre de la capacité souhaitée pour la prochaine activité de montée en puissance.
  - Pour les événements de mise à l'échelle horizontale, l'objectif est de procéder à une mise à l'échelle prudente afin de protéger la disponibilité de votre application, de sorte que les activités de mise à l'échelle horizontale sont bloquées jusqu'à l'expiration du temps de stabilisation. Toutefois, si une autre alarme lance une activité de montée en puissance au cours du temps de stabilisation de la diminution de charge, Application Auto Scaling monte immédiatement en puissance la cible. Dans ce cas, le temps de stabilisation de la mise à l'échelle horizontale s'arrête et ne se termine pas.
- Le planificateur de service respecte le nombre souhaité à tout moment mais, tant que vous avez des stratégies de mise à l'échelle et des alarmes actives sur un service, Service Auto Scaling peut modifier un nombre souhaité manuellement défini par vous-même.
- Si le nombre souhaité d’un service est défini comme inférieur à sa valeur de capacité minimale, et si une alarme déclenche une activité de montée en puissance, Service Auto Scaling augmente le nombre souhaité pour atteindre la valeur de capacité minimale, puis continue cette augmentation le cas échéant, selon la stratégie de mise à l’échelle associée à l’alarme. Toutefois, une activité de mise à l'échelle horizontale ne modifie pas le nombre souhaité, car il est déjà inférieur à la valeur de capacité minimale.
- Si le nombre souhaité d’un service est défini comme supérieur à sa valeur de capacité maximale, et si une alarme déclenche une activité de mise à l’échelle horizontale, Service Auto Scaling réduit le nombre souhaité pour atteindre la valeur de capacité maximale, puis continue cette réduction le

cas échéant, selon la stratégie de mise à l'échelle associée à l'alarme. Toutefois, une activité de montée en puissance ne modifie pas le nombre souhaité, car il est déjà supérieur à la valeur de capacité maximale.

- Au cours des activités de mise à l'échelle, le nombre réel de tâches en cours d'exécution dans un service correspond à la valeur utilisée par Service Auto Scaling comme point de départ, plutôt qu'au nombre souhaité. C'est ce qui est supposé représenter la capacité de traitement. Cela évite une mise à l'échelle excessive qui ne pourrait pas être satisfaite, lorsqu'il n'y a par exemple pas assez de ressources d'instances de conteneur pour placer les tâches supplémentaires. Si la capacité d'instance de conteneur est disponible ultérieurement, l'activité de mise à l'échelle en suspens peut réussir et d'autres activités de mise à l'échelle peuvent continuer après la période de stabilisation.
- Si vous souhaitez que votre nombre de tâches soit réduit à zéro lorsqu'il n'y a pas de travail à effectuer, définissez une capacité minimale de 0. Avec les stratégies de suivi des objectifs et d'échelonnement, lorsque la capacité réelle est de 0 et que la métrique indique qu'il y a une requête d'application, Service Auto Scaling attend l'envoi d'un point de données avant d'effectuer la montée en puissance. Dans ce cas, la mise à l'échelle entraîne la quantité la plus petite possible comme point de départ, puis reprend la mise à l'échelle en fonction du nombre réel de tâches en cours d'exécution.
- Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s'applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Pour plus d'informations, consultez Service Auto Scaling et déploiements.
- Vous disposez de plusieurs options d'Application Auto Scaling pour les tâches Amazon ECS. Le suivi des cibles est le mode le plus simple à utiliser. Il vous suffit de définir une valeur cible pour une métrique, telle que l'utilisation moyenne du processeur. Ensuite, l'autoscaler gère automatiquement le nombre de tâches nécessaires pour atteindre cette valeur. Grâce à la mise à l'échelle par étapes, vous pouvez réagir plus rapidement aux variations de la demande, car vous définissez les seuils spécifiques pour vos métriques de mise à l'échelle et le nombre de tâches à ajouter ou à supprimer lorsque les seuils sont dépassés. Et surtout, vous pouvez réagir très rapidement aux variations de la demande en réduisant la durée pendant laquelle une alarme de seuil est franchie.

Pour de plus amples informations sur les pratiques exemplaires en matière d'autoscaling de service, consultez la section Optimisation de l'autoscaling d'un service Amazon ECS.

Service Auto Scaling et déploiements

Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s’applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Si vous souhaitez suspendre les processus de montée en puissance pendant les déploiements, procédez comme suit.

1. Appelez la commande describe-scalable-targets, en spécifiant l'ID de ressource du service associé à la cible évolutive dans Application Auto Scaling (par exemple : `service/default/sample-webapp`). Enregistrez la sortie. Vous en aurez besoin pour appeler la commande suivante.
2. Appelez la commande register-scalable-target, en spécifiant l'ID de ressource, l'espace de noms et la dimension évolutive. Spécifiez `true` pour `DynamicScalingInSuspended` et `DynamicScalingOutSuspended`.
3. Une fois le déploiement terminé, vous pouvez appeler la commande register-scalable-target pour reprendre la mise à l'échelle.

Pour de plus amples informations, veuillez consulter la section relative à la Suspension et reprise de la mise à l'échelle pour Application Auto Scaling.

Utilisation d’une métrique cible pour mettre à l’échelle les services Amazon ECS

Grâce aux politiques de suivi des objectifs et d’échelonnement, vous sélectionnez une métrique et définissez une valeur cible. Amazon ECS Service Auto Scaling crée et gère CloudWatch des alarmes qui contrôlent la politique de mise à l'échelle et calculent l'ajustement de mise à l'échelle en fonction de la métrique et de la valeur cible. La politique de mise à l'échelle ajoute ou supprime des tâches de service si nécessaire pour maintenir la métrique à la valeur cible spécifiée ou proche de celle-ci. En plus de maintenir la métrique proche de la valeur cible, une politique de suivi des objectifs et d'échelonnement s'ajuste également aux fluctuations de la métrique dues à un modèle de charge fluctuant, et minimise les fluctuations rapides du nombre de tâches exécutées dans votre service.

Les politiques de suivi des cibles éliminent la nécessité de les définir manuellement CloudWatch alarmes et réglages de mise à l'échelle. Amazon ECS gère cela automatiquement en fonction de la cible que vous avez définie.

Tenez compte des éléments suivants lorsque vous utilisez des politiques de suivi des cibles :

- Une politique de suivi des objectifs et d'échelonnement suppose qu'elle doit effectuer une montée en puissance ; lorsque la métrique spécifiée est au-dessus de la valeur cible. Vous ne pouvez pas utiliser une politique de suivi des objectifs et d'échelonnement pour effectuer une montée en puissance lorsque la métrique spécifiée est en dessous de la valeur cible.
- Une politique de suivi des objectifs et d'échelonnement n'effectue pas de mise à l'échelle lorsque la métrique spécifiée a des données insuffisantes. Elle n'effectue pas de mise à l'échelle horizontale car elle n'interprète pas des données insuffisantes comme une faible utilisation.
- Vous pouvez constater des écarts entre la valeur cible et les points de données de métrique réels. Ceci est dû au fait que Service Auto Scaling agit toujours avec prudence en effectuant un arrondi vers le haut ou vers le bas quand il détermine la capacité à ajouter ou à enlever. Cela l'empêche d'ajouter une capacité insuffisante ou de retirer trop de capacité.
- Pour garantir la disponibilité de l'application, le service augmente proportionnellement aux métriques aussi rapidement que possible, mais diminue plus progressivement.
- Application Auto Scaling désactive les processus évolutifs pendant que les déploiements Amazon ECS sont en cours. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Ce comportement ne s’applique pas aux services Amazon ECS utilisant le contrôleur de déploiement externe. Pour plus d’informations, consultez Service Auto Scaling et déploiements.
- Vous pouvez avoir plusieurs politiques de suivi des objectifs et d'échelonnement pour un service Amazon ECS service dans la mesure où chacune d'elles utilise une métrique différente. L'objectif de Service Auto Scaling est de toujours donner la priorité à la disponibilité, afin que son comportement diffère selon que les politiques de suivi des objectifs et d'échelonnement sont prêtes pour une augmentation ou une diminution de taille. Il procèdera à la montée en puissance du service si l'une des politiques Suivi de la cible est prête pour une augmentation de taille, mais la diminuera uniquement si toutes les politiques Suivi de la cible (avec la portion de mise à l'échelle horizontale activée) sont prêtes pour une diminution de taille.
- Ne modifiez ni ne supprimez le CloudWatch alarmes gérées par Service Auto Scaling pour une politique de dimensionnement du suivi des cibles. Service Auto Scaling supprime les alarmes automatiquement quand vous supprimez la politique de mise à l'échelle.
- Le `ALBRequestCountPerTarget` la métrique pour les politiques de mise à l'échelle du suivi des cibles n'est pas prise en charge pour le blue/green type de déploiement.

Pour de plus amples informations sur les politiques de dimensionnement de suivi de cible, veuillez consulter Politiques de dimensionnement de suivi de cible dans le Manuel de l'utilisateur Application Auto Scaling.

Création d'une stratégie de mise à l'échelle du suivi de cible pour l'autoscaling d'un service Amazon ECS

Créez une stratégie de mise à l'échelle du suivi de cible pour qu'Amazon ECS augmente ou diminue automatiquement le nombre de tâches souhaité dans votre service. Le suivi de cible fonctionne à partir d'une valeur métrique cible.

Console

1. En plus des autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour l'autoscaling d'un service Amazon ECS.
2. Déterminez les métriques à utiliser pour la stratégie. Les mesures suivantes sont disponibles :
   - ECSServiceAverageCPUUtilization — L'utilisation moyenne du processeur que le service doit utiliser.
   - ECSServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire que le service doit utiliser.
   - ALBRequestCountPerTarget — Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.
3. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
4. Sur la page Clusters, choisissez le cluster.
5. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page de détails du service s'affiche.
6. Choisissez Définir le nombre de tâches.
7. Sous Nombre de tâches du service Amazon ECS, sélectionnez Utiliser l'autoscaling.

   La section Nombre de tâches s'affiche.

   a. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

   b. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

   c. Cliquez sur Enregistrer.

La page des stratégies s’affiche.

8. Choisissez Créer une stratégie de mise à l’échelle.

   La page Créer une stratégie s’affiche.

9. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible).
10. Pour Policy name (Nom de la politique), saisissez un nom de politique.
11. Sous Type de métrique, choisissez vos métriques dans la liste des options.
12. Pour Utilisation cible, saisissez la valeur cible pour le pourcentage de tâches qu’Amazon ECS doit gérer. L’autoscaling du service augmente horizontalement votre capacité jusqu’à ce que l’utilisation moyenne corresponde à l’objectif, ou jusqu’à ce qu’elle atteigne le nombre maximal de tâches que vous avez spécifié.
13. Sous Paramètres supplémentaires, procédez comme suit :

    a. Pour Scale-in période de refroidissement, entrez le délai en secondes qui s'écoule entre la fin d'une activité d'extension et le démarrage d'une autre activité d'extension.

    b. Pour Scale-out période de refroidissement, entrez le délai en secondes nécessaire pour qu'une activité de scale-out précédente prenne effet.

    c. Pour ne créer qu’une stratégie d’augmentation horizontale, sélectionnez Désactiver la réduction horizontale.

14. Choisissez Créer une stratégie de mise à l’échelle.

AWS CLI

1. Inscrivez le service Amazon ECS en tant que cible évolutive à l'aide de la commande register-scalable-target.
2. Créez une stratégie de mise à l’échelle à l’aide de la commande put-scaling-policy.

Mise à l'échelle automatique plus rapide grâce à des mesures haute résolution

Par défaut, Amazon ECS publie `CPUUtilization` et `MemoryUtilization` métriques à une résolution de 60 secondes. Vous pouvez configurer une résolution de 20 secondes pour accélérer la mise à l'échelle automatique des services.

Les mesures prédéfinies en haute résolution suivantes sont disponibles pour les politiques de dimensionnement du suivi des cibles :

PredefinedMetricType	Description
`ECSServiceAverageCPUUtilizationHighResolution`	Utilisation moyenne du processeur à une résolution de 20 secondes
`ECSServiceAverageMemoryUtilizationHighResolution`	Utilisation moyenne de la mémoire à une résolution de 20 secondes

Créez un nouveau service avec des mesures haute résolution pour une mise à l'échelle automatique plus rapide

Console

1. Sur le Clusters page, choisissez le cluster dans lequel vous souhaitez créer le service.
2. Dans le Des services section, choisissez Créez.
3. Dans le formulaire Créer un service, dans Configuration de surveillance section, choisissez Ajouter une configuration métrique.
4. Pour Nom de la métrique, sélectionnez la métrique que vous souhaitez collecter en haute résolution. Sélectionnez `CPUUtilization` ou `MemoryUtilization`.
5. Pour Résolution (secondes), sélectionnez 20.
6. (Facultatif) Pour ajouter les deux mesures en haute résolution, choisissez Ajouter une configuration métrique encore une fois et répétez les deux étapes précédentes pour l'autre métrique.
7. Dans le Mise à l'échelle automatique des services section, choisissez Utiliser le service Auto Scaling.
8. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l’autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
9. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser par l’autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

10. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible).
11. Pour Type métrique, sélectionnez la métrique haute résolution correspondante. Par exemple, choisissez `ECSServiceAverageCPUUtilizationHighResolution` si vous avez configuré `CPUUtilization` à une résolution de 20 secondes, ou `ECSServiceAverageMemoryUtilizationHighResolution` si vous avez configuré `MemoryUtilization` à une résolution de 20 secondes.
12. Pour Valeur cible, entrez le pourcentage d'utilisation cible. Le service Auto Scaling fait évoluer votre capacité jusqu'à ce que l'utilisation moyenne atteigne la valeur cible ou jusqu'à ce qu'elle atteigne le nombre maximum de tâches que vous avez spécifié.
13. En dessous Paramètres supplémentaires, configurez les éléments suivants :
    a. Pour Scale-out période de refroidissement, entrez le délai en secondes nécessaire pour qu'une activité de scale-out précédente prenne effet.
    b. Pour Scale-in période de refroidissement, entrez le délai en secondes qui s'écoule entre la fin d'une activité d'extension et le démarrage d'une autre activité d'extension.
14. Choisissez Créez pour créer le service.

AWS CLI

1. Créez le service avec la configuration de surveillance à l'aide du créer un service commande.

```
aws ecs create-service --region us-east-1 \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-app:1 \
    --desired-count 2 \
    --monitoring
 "metricConfigurations=[{metricNames=[CPUUtilization,MemoryUtilization],resolutionSeconds=2
```

2. Enregistrez le service en tant que cible évolutive à l'aide du register-scalable-target commande.

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --min-capacity 1 \
    --max-capacity 10 \
```

```
    --region us-east-1
```

3. Créez une politique de dimensionnement du suivi des cibles avec une métrique prédéfinie en haute résolution à l'aide du politique de mise à l'échelle de la production commande.

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --policy-name my-highres-cpu-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration '{
        "TargetValue": 50.0,
        "PredefinedMetricSpecification": {
            "PredefinedMetricType": "ECSServiceAverageCPUUtilizationHighResolution"
        }
    }' \
    --region us-east-1
```

4. Vérifiez que la politique a été créée à l'aide du décrire les politiques de mise à l'échelle commande.

```
aws application-autoscaling describe-scaling-policies \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --region us-east-1
```

Mettez à jour un service existant pour utiliser des métriques haute résolution pour une mise à l'échelle automatique plus rapide

La mise à jour d'un service existant afin d'utiliser des métriques haute résolution pour accélérer la mise à l'échelle automatique du service est un processus en deux étapes. Vous devez d'abord mettre à jour la résolution de surveillance, ce qui déclenche le déploiement d'un service. Une fois le déploiement terminé avec succès et toutes les tâches émettant des métriques à une résolution de 20 secondes, vous pouvez configurer une politique de dimensionnement haute résolution.

> **Important**
>
> Si votre service utilisait auparavant une résolution métrique de 60 secondes, les mesures haute résolution pour la politique de dimensionnement ne seront disponibles sur la console Amazon ECS qu'une fois que vous aurez mis à jour votre service pour utiliser une résolution métrique de 20 secondes et que le déploiement du service sera terminé.

Console

Étape 1 : mettre à jour la résolution de surveillance

1. Sur le Clusters page, choisissez le cluster qui contient le service que vous souhaitez mettre à jour.
2. Dans le Des services section, sélectionnez le service et choisissez Mettre à jour.
3. Dans le formulaire Service de mise à jour, dans Configuration de surveillance section, choisissez Ajouter une configuration métrique.
4. Pour Nom de la métrique, sélectionnez la métrique que vous souhaitez collecter en haute résolution. Sélectionnez `CPUUtilization` ou `MemoryUtilization`.
5. Pour Résolution (secondes), sélectionnez 20.
6. (Facultatif) Pour ajouter les deux mesures en haute résolution, choisissez Ajouter une configuration métrique encore une fois et répétez les deux étapes précédentes pour l'autre métrique.
7. Choisissez Mettre à jour pour soumettre la modification. Cela déclenche le déploiement d'un service.
8. Attendez la fin du déploiement. Vous pouvez suivre les progrès sur la page des détails du service dans Déploiements onglet.

> **Note**
>
> Les options de mesure de mise à l'échelle haute résolution sont grisées dans la section de mise à l'échelle automatique du service jusqu'à ce que le déploiement soit réussi et que toutes les tâches soient émises à une résolution de 20 secondes.

Étape 2 : configurer la politique de mise à l'échelle haute résolution

Une fois le déploiement terminé, configurez la politique de dimensionnement à l'aide de l'une des méthodes suivantes :

Option A : page de détails du service

1. Sur la page des détails du service, choisissez le Mise à l'échelle automatique des services onglet.
2. Choisissez Création d'une politique de dimensionnement (ou modifiez une politique existante).
3. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible).
4. Pour Type métrique, sélectionnez la métrique haute résolution. Par exemple, `ECSServiceAverageCPUUtilizationHighResolution` ou `ECSServiceAverageMemoryUtilizationHighResolution`.
5. Pour Valeur cible, entrez le pourcentage d'utilisation cible.
6. En dessous Paramètres supplémentaires, configurez les éléments suivants :
    a. Pour Scale-out période de refroidissement, entrez le délai en secondes nécessaire pour qu'une activité de scale-out précédente prenne effet.
    b. Pour Scale-in période de refroidissement, entrez le délai en secondes qui s'écoule entre la fin d'une activité d'extension et le démarrage d'une autre activité d'extension.
7. Choisissez Créer une stratégie de mise à l’échelle.

Option B : Mettre à jour le formulaire de service

1. Sur la page des détails du service, choisissez Mettre à jour.
2. Dans le Mise à l'échelle automatique des services section, choisissez Utiliser le service Auto Scaling (si ce n'est pas déjà fait).
3. Pour Nombre minimum de tâches, entrez la limite inférieure.
4. Pour Nombre maximum de tâches, entrez la limite supérieure.
5. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible).
6. Pour Type métrique, sélectionnez la métrique haute résolution (désormais activée une fois le déploiement terminé).

7. Pour Valeur cible, entrez le pourcentage d'utilisation cible.
8. En dessous Paramètres supplémentaires, réglez le Scale-out période de refroidissement et Scale-in période de refroidissement.
9. Choisissez Mettre à jour pour soumettre la modification.

AWS CLI

Étape 1 : mettre à jour la résolution de surveillance

1. Mettez à jour le service avec la configuration de surveillance à l'aide du service de mise à jour commande.

```
aws ecs update-service --region us-east-1 \
    --cluster my-cluster \
    --service my-service \
    --monitoring
 "metricConfigurations=[{metricNames=[CPUUtilization,MemoryUtilization],resolutionSeconds=2
```

2. Attendez la fin du déploiement. Vous pouvez suivre les progrès à l'aide du décrire les services commande et vérification de l'état du déploiement.

```
aws ecs describe-services --region us-east-1 \
    --cluster my-cluster \
    --services my-service
```

3. Une fois le déploiement terminé, vérifiez que la configuration de surveillance est active en décrivant la révision du service. Obtenez l'ARN de révision du service actif auprès du `describe-services` sortie, puis exécutez :

```
aws ecs describe-service-revisions --region us-east-1 \
    --service-revision-arns arn:aws:ecs:us-east-1:012345678910:service-revision/my-
cluster/my-service/1234567890123456789
```

Vérifiez que la configuration de surveillance apparaît dans la réponse :

```
{
    "serviceRevisions": [
        {
            ...
```

```
            "monitoring": {
                "metricConfigurations": [
                    {
                        "metricNames": ["CPUUtilization", "MemoryUtilization"],
                        "resolutionSeconds": 20
                    }
                ]
            }
        }
    ]
}
```

Étape 2 : configurer la politique de mise à l'échelle haute résolution

1. Enregistrez le service en tant que cible évolutive s'il n'est pas déjà enregistré, à l'aide du register-scalable-target commande.

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --min-capacity 1 \
    --max-capacity 10 \
    --region us-east-1
```

2. Créez une politique de dimensionnement du suivi des cibles avec une métrique prédéfinie en haute résolution à l'aide du politique de mise à l'échelle de la production commande. L'exemple suivant cible 50 % d'utilisation du processeur :

```
aws application-autoscaling put-scaling-policy \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --policy-name my-highres-cpu-policy \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration '{
        "TargetValue": 50.0,
        "PredefinedMetricSpecification": {
            "PredefinedMetricType": "ECSServiceAverageCPUUtilizationHighResolution"
        }
    }' \
```

```
    --region us-east-1
```

3. Vérifiez que la politique a été créée :

```
aws application-autoscaling describe-scaling-policies \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --region us-east-1
```

Validez un dimensionnement automatique plus rapide des services

Partie 1 : Vérifier que les métriques haute résolution sont publiées

1. Naviguez jusqu'au CloudWatch console.
2. Dans le volet de navigation, choisissez Métriques, puis choisissez Toutes les métriques.
3. Choisissez l’espace de nom AWS/ECS.
4. Choisissez ClusterName, ServiceName.
5. Localisez le `CPUUtilization` et `MemoryUtilization` des indicateurs pour votre service.
6. Réglez la période sur 20 secondes. Vous devriez voir des points de données toutes les 20 secondes, ce qui confirme que les mesures haute résolution sont actives.

Partie 2 : Vérifier que la politique de dimensionnement utilise la métrique haute résolution

1. Sur la page des détails du service, choisissez le Mise à l'échelle automatique des services onglet.
2. Consultez vos politiques de dimensionnement existantes et confirmez que la métrique est définie sur une variante haute résolution (`ECSServiceAverageCPUUtilizationHighResolution` ou `ECSServiceAverageMemoryUtilizationHighResolution`).
3. Vous pouvez également vérifier à l'aide de l'interface de ligne de commande :

```
aws application-autoscaling describe-scaling-policies \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --region us-east-1
```

Confirmez le `PredefinedMetricType` dans la réponse est définie sur une métrique haute résolution.

Considérations

- La surveillance des modifications de configuration crée une nouvelle révision du service et nécessite un déploiement de service. La métrique de mise à l'échelle haute résolution n'est pas disponible tant que le déploiement n'est pas terminé et que toutes les tâches sont émises à une résolution de 20 secondes.
- High-resolution les métriques sont prises en charge pour les services utilisant des équilibreurs de charge d'application et des équilibreurs de charge réseau. Les services utilisant des équilibreurs de charge classiques sans groupes cibles ne sont pas pris en charge.
- High-resolution les métriques ne sont pas prises en charge pour les services utilisant le `CODE_DEPLOY` ou `EXTERNAL` types de contrôleurs de déploiement.
- Vous pouvez consulter la configuration de surveillance de votre service à l'aide de `DescribeServiceRevisions`. La configuration de surveillance n'est pas renvoyée dans `CreateService` ou `UpdateService` réponses.

Utilisez des incréments prédéfinis en fonction de CloudWatch alarmes pour faire évoluer les services Amazon ECS

Grâce aux politiques de mise à l'échelle par étapes, vous créez et gérez CloudWatch alarmes qui invoquent le processus de dimensionnement. Lorsqu'une alarme est déclenchée, Amazon ECS lance la politique de mise à l'échelle associée à cette alarme. La stratégie de mise à l'échelle par étapes met à l'échelle les tâches à l'aide d'un ensemble d'ajustements, appelés ajustements par étapes. La taille de l'ajustement varie en fonction de l'ampleur du déclenchement de l'alarme.

- Si le dépassement dépasse le premier seuil, Amazon ECS applique le premier ajustement par étapes.
- Si le déclenchement dépasse le second seuil, Amazon ECS applique le deuxième ajustement par étapes, et ainsi de suite.

Nous vous recommandons fortement d'utiliser une stratégie de mise à l'échelle de suivi cible pour appliquer une mise à l'échelle en fonction de métriques telles que l'utilisation moyenne de l'UC ou le nombre moyen de requêtes par cible. Les métriques qui diminuent lorsque la capacité

augmente et augmentent lorsque la capacité diminue peuvent être utilisées pour augmenter ou réduire horizontalement le nombre de tâches proportionnellement en utilisant le suivi de cible. Cela permet de s'assurer qu'Amazon ECS suit de près la courbe de requêtes pour vos applications.

Création d'une stratégie de mise à l'échelle par étapes pour l'autoscaling de service Amazon ECS

Créez une stratégie de mise à l'échelle par étapes pour qu'Amazon ECS augmente ou diminue automatiquement le nombre souhaité de tâches dans votre service. La mise à l'échelle d'étape est basée sur un ensemble d'ajustements de mise à l'échelle, appelés ajustements par étapes, qui varient en fonction de l'importance du dépassement du seuil d'alarme.

Console

1. En plus des autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour l'autoscaling d'un service Amazon ECS.
2. Déterminez les métriques à utiliser pour la stratégie. Les mesures suivantes sont disponibles :
   - ECSServiceAverageCPUUtilization — L'utilisation moyenne du processeur que le service doit utiliser.
   - ECSServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire que le service doit utiliser.
   - ALBRequestCountPerTarget — Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.
3. Créez le CloudWatch alarmes pour les métriques. Pour plus d'informations, voir Créez un CloudWatch alarme basée sur un seuil statique dans le Amazon CloudWatch Guide de l'utilisateur.
4. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
5. Sur la page Clusters, choisissez le cluster.
6. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page de détails du service s'affiche.
7. Choisissez Définir le nombre de tâches.
8. Sous Nombre de tâches du service Amazon ECS, sélectionnez Utiliser l'autoscaling.

   La section Nombre de tâches s'affiche.

a. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

b. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

c. Cliquez sur Enregistrer.

La page des stratégies s'affiche.

9. Choisissez Créer une stratégie de mise à l'échelle.

La page Créer une stratégie s'affiche.

10. Pour Type de stratégie de mise à l'échelle, choisissez Mise à l'échelle par étapes.

11. Configurez les propriétés d'augmentation horizontale. Sous Étapes pour ajouter des tâches, procédez comme suit :

a. Pour Policy name (Nom de la politique), saisissez un nom de politique.

b. Pour CloudWatch nom de l'alarme, choisissez le CloudWatch alarme.

c. Pour Type d'agrégation des métriques, choisissez comment comparer la métrique sélectionnée au seuil défini.

d. Pour Types d'ajustement, choisissez si l'ajustement est basé sur une modification du nombre de tâches ou sur une modification du pourcentage de tâches.

e. Dans Actions à entreprendre, saisissez les valeurs correspondant à l'action à entreprendre.

Choisissez Ajouter une étape pour ajouter des actions supplémentaires.

12. Configurez les propriétés de réduction horizontale. Sous Étapes pour supprimer des tâches, procédez comme suit :

a. Pour Policy name (Nom de la politique), saisissez un nom de politique.

b. Pour CloudWatch nom de l'alarme, choisissez le CloudWatch alarme.

c. Pour Type d'agrégation des métriques, choisissez comment comparer la métrique sélectionnée au seuil défini.

d. Pour Types d'ajustement, choisissez si l'ajustement est basé sur une modification du nombre de tâches ou sur une modification du pourcentage de tâches.

e. Dans Actions à entreprendre, saisissez les valeurs correspondant à l'action à entreprendre.

Choisissez Ajouter une étape pour ajouter des actions supplémentaires.

13. Pour Temps de stabilisation, saisissez la durée, en secondes, à attendre avant qu'une activité de mise à l'échelle précédente ne prenne effet. Pour une stratégie d'ajout, il s'agit du délai après une activité d'augmentation horizontale pendant lequel la stratégie de mise à l'échelle bloque les activités de réduction horizontale et limite le nombre de tâches pouvant augmenter horizontalement simultanément. Pour une politique de suppression, il s'agit du délai qui doit s'écouler après une activité de réduction horizontale avant qu'une autre activité de réduction puisse commencer.
14. Choisissez Créer une stratégie de mise à l'échelle.

AWS CLI

1. Inscrivez le service Amazon ECS en tant que cible évolutive à l'aide de la commande register-scalable-target.
2. Créez une stratégie de mise à l'échelle à l'aide de la commande put-scaling-policy.

Utilisation des actions planifiées pour mettre à l'échelle les services Amazon ECS

Avec la mise à l'échelle planifiée, vous pouvez configurer la mise à l'échelle automatique de votre application en fonction des changements de charge prévisibles en créant des actions planifiées qui augmentent ou diminuent le nombre de tâches à des moments précis. Cette procédure vous permet de mettre votre application à l'échelle de manière proactive afin qu'elle corresponde aux variations de charge prévisibles.

Ces actions de mise à l'échelle planifiées vous permettent d'optimiser les coûts et les performances. Votre application dispose d'un nombre suffisant de tâches pour gérer le pic de trafic en milieu de semaine, mais ne surdimensionne pas le nombre de tâches à d'autres moments.

Vous pouvez utiliser conjointement la mise à l'échelle planifiée et les stratégies de mise à l'échelle pour bénéficier des avantages des approches proactives et réactives de la mise à l'échelle. Après l'exécution d'une action de mise à l'échelle planifiée, la stratégie de mise à l'échelle peut continuer à prendre des décisions sur l'opportunité de poursuivre la mise à l'échelle du nombre de tâches. Cela vous permet de vous assurer que vous avez un nombre de tâches suffisant pour gérer la charge de votre application. Bien que votre application soit mise à l'échelle pour répondre à la demande, la capacité actuelle doit se situer dans les limites du nombre de tâches minimal et maximal qui a été fixée par votre action planifiée.

Vous pouvez configurer la mise à l'échelle du calendrier à l'aide de l' AWS CLI. Pour de plus amples informations sur l'utilisation de la mise à l'échelle planifiée, consultez la section Mise à l'échelle planifiée dans le Guide de l'utilisateur Application Auto Scaling.

Création d'une action planifiée pour l'autoscaling d'un service Amazon ECS

Créez une action planifiée pour qu'Amazon ECS augmente ou diminue le nombre de tâches exécutées par votre service en fonction de la date et de l'heure.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page de détails du service s'affiche.
4. Choisissez Autoscaling du service.

   La page d'autoscaling du service s'affiche.
5. Si vous n'avez pas configuré l'autoscaling du service, sélectionnez Définir le nombre de tâches.

   La section Nombre de tâches du service Amazon ECS s'affiche.

   Sous Nombre de tâches du service Amazon ECS, choisissez Utiliser l'autoscaling du service pour ajuster le nombre de tâches souhaité pour votre service.

   La section Nombre de tâches s'affiche.

   a. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
   b. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.
   c. Choisissez Enregistrer.

      La page des stratégies s'affiche.
6. Choisissez Actions planifiées, puis Créer.

   La page Créer une action planifiée s'affiche.

7. Pour Nom de l'action, saisissez un nom unique.
8. Dans le champ Fuseau horaire, choisissez un fuseau horaire.

   Tous les fuseaux horaires répertoriés proviennent de la base de données des fuseaux horaires IANA. Pour plus d'informations, consultez la section Liste des fuseaux horaires de la base de données tz.
9. Pour Heure de début, saisissez la date et l'heure de début de l'action.

   Si vous avez choisi une planification récurrente, l'heure de début définit le moment où la première action planifiée de la série récurrente s'exécute.
10. Pour Recurrence (Récurrence), choisissez l'une des options disponibles.

    - Pour effectuer des mises à l'échelle selon un calendrier récurrent, choisissez la fréquence à laquelle Amazon ECS exécute l'action planifiée.
      - Si vous sélectionnez une option qui commence par Fréquence, l'expression Cron est créée pour vous.
      - Si vous sélectionnez Cron, tapez une expression Cron qui spécifie l'heure à laquelle exécuter l'action.
    - Pour ne mettre à l'échelle qu'une seule fois, choisissez Une fois.
11. Sous Ajustements des tâches, procédez comme suit :

    - Pour Minimum, saisissez le nombre minimum de tâches que le service doit exécuter.
    - Pour Maximum, saisissez le nombre maximum de tâches que le service doit exécuter.
12. Choisissez Create Scheduled Action (Créer une action planifiée).

INTERFACE DE LIGNE DE COMMANDE (CLI)

Utilisez le AWS CLI comme suit pour configurer des politiques de dimensionnement planifiées pour votre service. Remplacez chaque *user input placeholder* par vos propres informations.

Exemple : Pour planifier un dimensionnement unique

Utilisez la commande put-scheduled-action suivante avec l'option `--start-time "YYYY-MM-DDThh:mm:ssZ"` et l'une ou les deux options `--MinCapacity` et `--MaxCapacity`.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-one-time-schedule \
```

```
  --start-time 2021-01-30T12:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Exemple : pour planifier la mise à l'échelle selon un calendrier récurrent

Utilisez la commande put-scheduled-action suivante : Remplacez le *user input* avec tes valeurs.

```
aws application-autoscaling put-scheduled-action --service-namespace ecs \
  --resource-id service/my-cluster/my-service \
  --scheduled-action-name my-recurring-action \
  --schedule "rate(5 hours)" \
  --start-time 2021-01-30T12:00:00 \
  --end-time 2021-01-31T22:00:00 \
  --scalable-target-action MinCapacity=3,MaxCapacity=10
```

Le calendrier de récurrence spécifié est basé sur le fuseau horaire UTC. Pour spécifier un fuseau horaire différent, incluez l'option `--time-zone` et le nom du fuseau horaire IANA, comme dans l'exemple suivant.

```
--time-zone "America/New_York"
```

Pour plus d'informations, consultez la section Liste des fuseaux horaires de la base de données tz.

Utilisation des tendances historiques pour mettre à l'échelle les services Amazon ECS via une mise à l'échelle prédictive

La mise à l'échelle prédictive examine les données de charge passées issues des flux de trafic afin d'analyser les tendances quotidiennes ou hebdomadaires. Elle utilise ensuite cette analyse pour anticiper les besoins postérieurs et augmenter de manière proactive les tâches dans votre service selon les besoins.

L'autoscaling prédictif est particulièrement utile dans les cas suivants.

- Trafic cyclique : utilisation accrue des ressources pendant les heures normales de bureau et utilisation réduite pendant les soirées et les fins de semaine.
- Modèles de charge de travail récurrents et intermittents : par exemple, traitement par lots, tests ou analyse périodique des données.
- Applications avec des temps d'initialisation longs : cette situation peut avoir un impact sur les performances des applications lors d'événements d'augmentation horizontale, entraînant une latence notable.

Si vos applications prennent beaucoup de temps à s'initialiser et que le trafic augmente de manière régulière, vous devriez envisager d'utiliser la mise à l'échelle prédictive. Elle permet d'accélérer la mise à l'échelle en augmentant de manière proactive le nombre de tâches pour les charges prévues, au lieu d'utiliser uniquement des stratégies de mise à l'échelle dynamique, telles que la stratégie de mise à l'échelle du suivi de cible ou la mise à l'échelle par étapes. En vous aidant à éviter le risque de surallocation du nombre de tâches, la mise à l'échelle prédictive peut également vous permettre de réaliser des économies.

Prenons l'exemple d'une application très utilisée pendant les heures de bureau et peu utilisée la nuit. Au début de chaque journée ouvrable, la mise à l'échelle prédictive peut augmenter horizontalement les tâches avant le premier afflux de trafic. Cela permet à votre application de maintenir une disponibilité et des performances élevées lorsqu'elle passe d'une période de faible utilisation à une période d'utilisation plus intensive. Vous n'avez pas besoin d'attendre que la mise à l'échelle dynamique réagisse à l'évolution du trafic. Vous n'avez pas non plus à passer du temps à examiner les modèles de charge de votre application et à essayer de planifier le nombre adéquat de tâches à l'aide d'une mise à l'échelle planifiée.

La mise à l'échelle prédictive est une fonctionnalité au niveau du service qui met à l'échelle la tâche de votre service indépendamment de la mise à l'échelle de la capacité de calcul sous-jacente (par exemple, EC2 ou Fargate). Pour Fargate, AWS gère et adapte automatiquement la capacité sous-jacente en fonction des exigences des tâches. Pour la capacité EC2, vous pouvez utiliser les fournisseurs de capacité du groupe Auto Scaling pour mettre automatiquement à l'échelle les instances EC2 sous-jacentes en fonction des exigences de dimensionnement de vos tâches.

Table des matières

- Fonctionnement de la mise à l'échelle prédictive dans Amazon ECS
- Création d'une stratégie de mise à l'échelle pour l'autoscaling d'un service Amazon ECS
- Évaluation de vos stratégies de mise à l'échelle prédictive pour Amazon ECS
- Utilisation des actions planifiées pour remplacer les valeurs prévisionnelles pour Amazon ECS
- Stratégie avancée de mise à l'échelle prédictive utilisant des métriques personnalisées pour Amazon ECS

Fonctionnement de la mise à l'échelle prédictive dans Amazon ECS

Vous trouverez ici des informations sur les considérations relatives à l'utilisation de la mise à l'échelle prédictive, son fonctionnement et ses limites.

Considérations relatives à l’utilisation de la mise à l’échelle prédictive

- Vous voulez vous assurer que la mise à l’échelle prédictive est adaptée à votre charge de travail. Vous pouvez vérifier cela en configurant les stratégies de mise à l’échelle en mode prévisions uniquement et en consultant les recommandations de la console. Vous devez évaluer les prévisions et les recommandations avant de commencer à utiliser la mise à l’échelle prédictive.
- Avant que la mise à l’échelle prédictive puisse commencer à établir des prévisions, elle a besoin d’au moins 24 heures de données historiques. Plus il y a de données historiques disponibles, plus les prévisions sont efficaces, l’idéal étant deux semaines. Vous devrez également attendre 24 heures avant que la mise à l’échelle prédictive puisse générer de nouvelles prévisions lorsque vous supprimez un service Amazon ECS pour en recréer un. L’une des manières d’accélérer les choses consiste à utiliser des métriques personnalisées pour agréger les métriques entre l’ancien et le nouveau service Amazon ECS.
- Choisissez une métrique de charge qui représente avec précision la charge complète de votre application et qui constitue l’aspect le plus important de votre application à prendre en compte.
- La mise à l’échelle dynamique avec mise à l’échelle prédictive vous permet de suivre de près la demande de votre application, afin que vous puissiez l’augmenter en période d’accalmie et l’augmenter en cas d’augmentation inattendue du trafic. Lorsque plusieurs stratégies de mise à l’échelle sont actives, chaque stratégie détermine indépendamment le nombre souhaité de tâches, et le nombre souhaité de tâches est défini sur le maximum de celles-ci.
- Vous pouvez utiliser la mise à l’échelle prédictive parallèlement à vos stratégies de mise à l’échelle dynamique, telles que le suivi des cibles ou la mise à l’échelle par étapes, afin que vos applications évoluent en fonction de modèles historiques et en temps réel. En soi, la mise à l’échelle prédictive ne permet pas de réduire horizontalement vos tâches.
- Si vous utilisez un rôle personnalisé lorsque vous appelez l’API `register-scalable-target`, un message d’erreur peut s’afficher indiquant que la stratégie de mise à l’échelle prédictive ne peut fonctionner que si le SLR est activé. Dans ce cas, vous devez appeler `register-scalable-target` à nouveau, mais sans role-arn. Utilisez le SLR lors de l’enregistrement de la cible évolutive et appelez l’API `put-scaling-policy`.

Fonctionnement de la mise à l'échelle prédictive

Vous utilisez la mise à l'échelle prédictive en créant une politique de mise à l'échelle prédictive qui spécifie CloudWatch métrique à surveiller et à analyser. La mise à l’échelle prédictive doit disposer d’au moins 24 heures de données pour commencer à prévoir les valeurs futures.

Une fois que vous avez créé la stratégie, la mise à l'échelle prédictive commence à analyser les données métriques recueillies au cours des 14 derniers jours afin d'identifier des modèles. Cette analyse est utilisée pour générer des prévisions horaires des besoins pour les 48 prochaines heures. Le plus récent CloudWatch les données sont utilisées pour mettre à jour les prévisions toutes les six heures. À mesure que de nouvelles données arrivent, la mise à l'échelle prédictive améliore continuellement la précision des prévisions futures.

Lorsque vous activez la mise à l'échelle prédictive pour la première fois, elle s'exécute en mode prévision uniquement. Elle génère des prévisions dans ce mode, mais ne met pas votre service Amazon ECS à l'échelle en fonction de ces prévisions. Cela vous permet d'évaluer la précision et la pertinence des prévisions. Vous pouvez consulter les données de prévision à l'aide de l'opération API `GetPredictiveScalingForecast` ou de l' Console de gestion AWS.

Lorsque vous décidez de commencer à utiliser la mise à l'échelle prédictive, passez la stratégie de mise à l'échelle en mode prévision et mise à l'échelle. Dans ce mode, les événements suivants se produisent.

Votre service Amazon ECS est mis à l'échelle au début de chaque heure en fonction des prévisions pour cette heure, par défaut. Vous pouvez choisir de commencer plus tôt en utilisant la propriété `SchedulingBufferTime` dans l'opération d'API `PutScalingPolicy`. Cela permet de lancer de nouvelles tâches avant la demande prévue et leur donne le temps de démarrer et de se préparer à gérer le trafic.

Limite maximum de tâches

Lorsque vous enregistrez les services Amazon ECS pour la mise à l'échelle, vous définissez un nombre maximal de tâches pouvant être lancées par service. Par défaut, lorsque des politiques de mise à l'échelle sont définies, elles ne peuvent pas augmenter le nombre de tâches au-dessus de sa limite maximale.

Vous pouvez également autoriser l'augmentation automatique du nombre maximum de tâches du service si les prévisions approchent ou dépassent le nombre maximum de tâches du service Amazon ECS.

> **Warning**
>
> Soyez prudent lorsque vous autorisez l'augmentation automatique du nombre maximum de tâches. Cela peut entraîner le lancement d'un plus grand nombre de tâches que prévu, si le nombre maximum de tâches augmenté n'est pas surveillé et géré. Le nombre maximum

de tâches augmenté devient alors le nouveau nombre maximum normal de tâches pour le service Amazon ECS jusqu'à ce que vous le mettiez à jour manuellement. Le nombre maximum de tâches ne revient pas automatiquement au maximum initial.

Régions prises en charge

- 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 (Malaisie)
- Asie-Pacifique (Melbourne)
- Asie-Pacifique (Mumbai)
- Asie-Pacifique (Nouvelle Zélande)
- Asie-Pacifique (Osaka)
- Asie-Pacifique (Séoul)
- Asie-Pacifique (Singapour)
- Asie-Pacifique (Sydney)
- Asie-Pacifique (Taipei)
- Asie-Pacifique (Thaïlande)
- 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)
- Mexique (Centre)
- Middle East (Bahrain)
- Amérique du Sud (São Paulo)
- AWS GovCloud (US-East)
- AWS GovCloud (US-West)

Création d'une stratégie de mise à l'échelle pour l'autoscaling d'un service Amazon ECS

Créez une stratégie de mise à l'échelle prédictive pour qu'Amazon ECS augmente ou diminue le nombre de tâches exécutées par votre service en fonction de données historiques.

> **Note**
>
> Un nouveau service doit fournir au moins 24 heures de données avant de pouvoir générer une prévision.

Console

1. En plus des autorisations IAM standard pour créer et mettre à jour des services, vous avez besoin d'autorisations supplémentaires. Pour plus d'informations, consultez Autorisations IAM requises pour l'autoscaling d'un service Amazon ECS.
2. Déterminez les métriques à utiliser pour la stratégie. Les mesures suivantes sont disponibles :
   - ECSServiceAverageCPUUtilization — L'utilisation moyenne du processeur que le service doit utiliser.

- ECSServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire que le service doit utiliser.
- ALBRequestCountPerTarget — Le nombre moyen de demandes par minute que cette tâche devrait idéalement recevoir.

Vous pouvez également utiliser une métrique personnalisée. Vous devez définir les valeurs suivantes :

- Charge : métrique qui représente avec précision la charge totale de votre application et qui constitue l'aspect le plus important à prendre en compte pour la mise à l'échelle de votre application.

- Métrique de mise à l'échelle : le meilleur indicateur pour déterminer le niveau d'utilisation idéal pour votre application.

3. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
4. Sur la page Clusters, choisissez le cluster.
5. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page de détails du service s'affiche.
6. Choisissez Autoscaling du service, puis sélectionnez Définir le nombre de tâches.
7. Sous Nombre de tâches du service Amazon ECS, sélectionnez Utiliser l'autoscaling.

   La section Nombre de tâches s'affiche.

   a. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.
   b. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.
   c. Cliquez sur Enregistrer.

      La page des stratégies s'affiche.
8. Choisissez Créer une stratégie de mise à l'échelle.

   La page Créer une stratégie s'affiche.
9. Pour Type de stratégie de mise à l'échelle, choisissez Mise à l'échelle prédictive.

10. Pour Policy name (Nom de la politique), saisissez un nom de politique.
11. Sous Paire de métriques, choisissez vos métriques dans la liste des options.

    Si vous avez choisi Nombre de requêtes Application Load Balancer par cible, choisissez un groupe cible dans le champ Groupe cible. L'option Nombre de requêtes Application Load Balancer par cible n'est prise en charge que si vous avez attaché un groupe cible Application Load Balancer à votre service.

    Si vous avez choisi Paire de métriques personnalisées, sélectionnez les métriques individuelles dans les listes déroulantes Métrique de charge et Métrique de mise à l'échelle.
12. Pour Utilisation cible, saisissez la valeur cible pour le pourcentage de tâches qu'Amazon ECS doit gérer. L'autoscaling du service augmente horizontalement votre capacité jusqu'à ce que l'utilisation moyenne corresponde à l'objectif, ou jusqu'à ce qu'elle atteigne le nombre maximal de tâches que vous avez spécifié.
13. Choisissez Créer une stratégie de mise à l'échelle.

AWS CLI

Utilisez le AWS CLI comme suit pour configurer des politiques de dimensionnement prédictif pour votre service Amazon ECS. Remplacez chaque *user input placeholder* par vos propres informations.

Pour plus d'informations sur le CloudWatch mesures que vous pouvez spécifier, voir PredictiveScalingMetricSpecification dans le Référence de l'API Amazon EC2 Auto Scaling.

Exemple 1 : stratégie de mise à l'échelle prédictive avec mémoire prédéfinie.

Voici un exemple de stratégie avec une configuration mémoire prédéfinie.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 40,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceMemoryUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
```

```
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
}
```

L'exemple suivant illustre la création de la stratégie en exécutant la commande put-scaling-policy avec le fichier de configuration spécifié.

```
aws application-autoscaling put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Si elle aboutit, cette commande renvoie l'ARN de la stratégie.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Exemple 2 : stratégie de mise à l'échelle prédictive avec UC prédéfinie.

Voici un exemple de stratégie avec une configuration UC prédéfinie.

```
cat policy.json
{
    "MetricSpecifications": [
        {
            "TargetValue": 0.00000004,
            "PredefinedMetricPairSpecification": {
                "PredefinedMetricType": "ECSServiceCPUUtilization"
            }
        }
    ],
    "SchedulingBufferTime": 3600,
    "MaxCapacityBreachBehavior": "HonorMaxCapacity",
    "Mode": "ForecastOnly"
```

```
}
```

L'exemple suivant illustre la création de la stratégie en exécutant la commande put-scaling-policy avec le fichier de configuration spécifié.

```
aws aas put-scaling-policy \
--service-namespace ecs \
--region us-east-1 \
--policy-name predictive-scaling-policy-example \
--resource-id service/MyCluster/test \
--policy-type PredictiveScaling \
--scalable-dimension ecs:service:DesiredCount \
--predictive-scaling-policy-configuration file://policy.json
```

Si elle aboutit, cette commande renvoie l'ARN de la stratégie.

```
{
    "PolicyARN": "arn:aws:autoscaling:us-
east-1:012345678912:scalingPolicy:d1d72dfe-5fd3-464f-83cf-824f16cb88b7:resource/ecs/
service/MyCluster/test:policyName/predictive-scaling-policy-example",
    "Alarms": []
}
```

Évaluation de vos stratégies de mise à l'échelle prédictive pour Amazon ECS

Avant d'utiliser une stratégie de mise à l'échelle prédictive pour mettre à l'échelle vos services, consultez les recommandations et autres données relatives à votre stratégie dans la console Amazon ECS. C'est une étape importante pour éviter qu'une politique de mise à l'échelle prédictive mette votre capacité réelle à l'échelle tant que vous ne savez pas si ses prévisions sont exactes.

Si le service est neuf, prévoyez 24 heures pour créer la première prévision.

Quand AWS crée une prévision, elle utilise des données historiques. Si votre service ne dispose pas encore de beaucoup de données historiques récentes, la mise à l'échelle prédictive peut temporairement compléter les prévisions avec des agrégats créés à partir des agrégats historiques actuellement disponibles. Les prévisions sont remplies jusqu'à deux semaines avant la date de création d'une politique.

Afficher vos recommandations de mise à l'échelle prédictive

Pour une analyse efficace, l'autoscaling du service doit disposer d'au moins deux stratégies de mise à l'échelle prédictive à comparer. (Toutefois, vous pouvez toujours consulter les résultats d'une seule

politique.) Lorsque vous créez plusieurs politiques, vous pouvez évaluer une politique qui utilise une seule métrique par rapport à une autre qui utilise une autre métrique. Vous pouvez également évaluer l'impact de différentes combinaisons de valeurs cibles et de métriques. Une fois les stratégies de mise à l’échelle prédictive créées, Amazon ECS commence immédiatement à évaluer la stratégie la plus appropriée pour mettre votre groupe à l’échelle.

Pour afficher vos recommandations dans la console Amazon ECS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page de détails du service s’affiche.
4. Choisissez Autoscaling du service.
5. Choisissez la stratégie de mise à l’échelle prédictive, puis sélectionnez Actions, Mise à l’échelle prédictive, Afficher les recommandations.

   Vous pouvez consulter les détails d’une stratégie ainsi que notre recommandation. La recommandation vous indique si la politique de mise à l'échelle prédictive est plus efficace que si vous ne l'utilisez pas.

   Si vous ne savez pas si une politique de mise à l'échelle prédictive convient à votre groupe, consultez les colonnes Impact sur la disponibilité et Impact sur les coûts pour choisir la politique appropriée. Les informations de chaque colonne vous indiquent l'impact de la politique.

   - Impact sur la disponibilité : indique si la stratégie permettrait d’éviter un impact négatif sur la disponibilité en provisionnant suffisamment d’instances pour gérer la charge de travail, en comparaison avec sa non-utilisation.
   - Impact sur les coûts : indique si la stratégie éviterait un impact négatif sur vos coûts en évitant une surallocation des tâches, en comparaison avec sa non-utilisation. En cas d’allocation excessive, vos services sont sous-utilisés ou inactifs, ce qui ne fait qu’augmenter l’impact sur les coûts.

   Si vous avez plusieurs politiques, la balise Meilleure prévision s'affiche à côté du nom de la politique qui offre le plus d'avantages en matière de disponibilité à moindre coût. Une plus grande importance est accordée à l'impact sur la disponibilité.

6. (Facultatif) Pour sélectionner la période souhaitée pour les résultats des recommandations, choisissez la valeur de votre choix dans le menu déroulant Période d'évaluation : 2 jours, 1 semaine ou 2 semaines. Par défaut, la période d'évaluation est réglée sur les deux dernières semaines. Une période d'évaluation plus longue fournit davantage de points de données pour les résultats des recommandations. Toutefois, l'ajout de points de données supplémentaires risque de ne pas améliorer les résultats si vos modèles de charge ont changé, par exemple après une période de demande exceptionnelle. Dans ce cas, vous pouvez obtenir une recommandation plus ciblée en consultant des données plus récentes.

> **Note**
>
> Les recommandations sont générées uniquement pour les politiques qui sont en mode Prévision uniquement. La fonctionnalité des recommandations offre de meilleurs résultats lorsqu'une politique est en mode Prévision uniquement pendant toute la période d'évaluation. Si vous lancez une politique en mode Prévision et mise à l'échelle et que vous la passez ultérieurement en mode Prévision uniquement, les résultats de cette politique risquent d'être biaisés. Cela s'explique par le fait que la politique a déjà contribué à la capacité réelle.

Consulter les graphiques de surveillance de la mise à l'échelle prédictive

Dans la console, vous pouvez consulter les prévisions des jours, semaines ou mois précédents afin de visualiser les performances de la stratégie au fil du temps. Vous pouvez également utiliser ces informations pour évaluer la précision des prévisions lorsque vous décidez de laisser une stratégie mettre à l'échelle votre nombre réel de tâches.

Pour consulter les graphiques de surveillance de la mise à l'échelle prédictive dans la console Amazon ECS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page de détails du service s'affiche.
4. Choisissez Autoscaling du service.
5. Choisissez la stratégie de mise à l'échelle prédictive, puis sélectionnez Actions, Mise à l'échelle prédictive, Afficher les graphiques.

6. Dans la section Surveillance, vous pouvez afficher les prévisions passées et futures de votre politique concernant la charge et la capacité par rapport aux valeurs réelles. Le graphique Charge présente la prévision de charge et les valeurs réelles pour la métrique de charge que vous avez choisie. Le graphique Capacité indique le nombre de tâches prédit par la stratégie. Il inclut également le nombre réel de tâches lancées. La ligne verticale sépare les valeurs historiques des prévisions futures. Ces graphiques sont disponibles peu de temps après la création de la politique.
7. (Facultatif) Pour modifier la quantité de données historiques affichées dans le graphique, choisissez la valeur de votre choix dans Période d'évaluation liste déroulante. La période d'évaluation ne transforme en rien les données de cette page. Elle ne fait que modifier la quantité de données historiques affichées.

Comparez les données du Equilibreur de graphe

Chaque ligne horizontale représente un ensemble différent de points de données rapportés à des intervalles d'une heure :

1. La charge observée réelle utilise la statistique SUM correspondant à la métrique de charge de votre choix afin d'afficher la charge horaire totale dans le passé.
2. La charge prévue par la politique indique la prévision de charge horaire. Cette prévision se base sur les observations de charge réelles des deux semaines précédentes.

Comparez les données du Capacity graphe

Chaque ligne horizontale représente un ensemble différent de points de données rapportés à des intervalles d'une heure :

1. Le nombre réel de tâches observé indique la capacité réelle de votre service Amazon ECS dans le passé, qui dépend de vos autres stratégies de mise à l’échelle et de la taille minimale du groupe en vigueur pour la période sélectionnée.
2. La capacité prévue par la politique indique la capacité de base à laquelle vous pouvez vous attendre au début de chaque heure lorsque la politique est en mode Prévision et mise à l'échelle.
3. Le nombre de tâches requis estimé indique le nombre idéal de tâches dans votre service pour maintenir la métrique de mise à l’échelle à la valeur cible que vous avez choisie.
4. Le nombre minimum de tâches indique le nombre minimum de tâches de votre service.
5. La capacité maximale indique le nombre maximal de tâches de votre service.

Afin de calculer la capacité requise estimée, nous partons du principe que chaque tâche est utilisée de manière égale à une valeur cible spécifiée. Dans la pratique, le nombre de tâches n'est pas utilisé de manière égale. En supposant toutefois que l'utilisation est répartie uniformément entre les tâches, nous pouvons estimer la probabilité de la capacité nécessaire. Le nombre de tâches requis est alors calculé comme étant inversement proportionnel à la métrique de mise à l'échelle que vous avez utilisée pour votre stratégie de mise à l'échelle prédictive. En d'autres termes, à mesure que le nombre de tâches augmente, la métrique de mise à l'échelle diminue au même rythme. Par exemple, si le nombre de tâches double, la métrique de mise à l'échelle doit diminuer de moitié.

La formule de la capacité requise déduite est la suivante :

```
sum of (actualServiceUnits*scalingMetricValue)/(targetUtilization)
```

Par exemple, nous prenons les `actualServiceUnits` (`10`) et la `scalingMetricValue` (`30`) pour une heure donnée. Nous prenons ensuite la `targetUtilization` que vous avez spécifiée dans votre politique de mise à l'échelle prédictive (`60`) et calculons la capacité requise déduite pour la même heure. Elle renvoie la valeur `5`. Cela signifie que cinq est la quantité de capacité requise déduite pour maintenir la capacité en proportion inverse directe à la valeur cible de la métrique de mise à l'échelle.

> **Note**
>
> Différents leviers sont à votre disposition pour ajuster et améliorer les économies de coûts et la disponibilité de votre application.
>
> - Vous utilisez la mise à l'échelle prédictive pour la capacité de base et la mise à l'échelle dynamique afin de gérer la capacité supplémentaire. La mise à l'échelle dynamique fonctionne indépendamment de la mise à l'échelle prédictive, avec une mise à l'échelle horizontale et une montée en puissance en fonction de l'utilisation actuelle. Tout d'abord, Amazon ECS calcule le nombre de tâches recommandé pour chaque stratégie de mise à l'échelle non planifiée. Ensuite, il procède à la mise à l'échelle en fonction de la stratégie qui fournit le plus grand nombre de tâches.
> - Pour permettre la réduction horizontale lorsque la charge diminue, votre service doit toujours disposer d'au moins une stratégie de mise à l'échelle dynamique avec la partie réduction horizontale activée.
> - Vous pouvez améliorer les performances de mise à l'échelle en vous assurant que vos capacités minimale et maximale ne sont pas trop restrictives. Une stratégie avec un

nombre recommandé de tâches qui ne se situe pas dans la plage de capacité minimale et maximale ne pourra pas être mise à l'échelle.

Surveillez les métriques de dimensionnement prédictif pour Amazon ECS avec CloudWatch

Vous pouvez utiliser Amazon CloudWatch pour surveiller vos données à des fins de dimensionnement prédictif. Une stratégie de mise à l'échelle prédictive collecte des données qui sont utilisées pour prévoir votre charge postérieure. Les données collectées sont automatiquement stockées dans CloudWatch à intervalles réguliers et peut être utilisé pour visualiser les performances de la politique au fil du temps. Vous pouvez également créer CloudWatch alarmes pour vous avertir lorsque les indicateurs de performance changent au-delà des limites que vous avez définies.

Visualisez les données de prévision historiques

Les données de prévision de charge pour une politique de dimensionnement prédictif peuvent être consultées dans CloudWatch et peut être utile pour visualiser les prévisions par rapport à d'autres CloudWatch métriques dans un seul graphique. Vous pouvez également observer les tendances au fil du temps en affichant une période plus longue. Vous pouvez accéder aux métriques historiques jusqu'à 15 mois pour acquérir un meilleur point de vue de la façon dont votre politique s'exécute.

Pour consulter les données prévisionnelles historiques à l'aide du CloudWatch console

1. Ouvrez le CloudWatch console à https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Metrics (Métriques), All metrics (Toutes les métriques).
3. Choisissez l'espace de noms de métrique Application Auto Scaling.
4. Choisissez Prévisions de charge pour la mise à l'échelle prédictive.
5. Dans le champ de recherche, saisissez le nom de la stratégie de mise à l'échelle prédictive ou le nom du groupe de services Amazon ECS, puis appuyez sur « Entrée » pour filtrer les résultats.
6. Pour représenter graphiquement une métrique, cochez la case en regard de la métrique. Pour modifier le nom du graphique, choisissez l'icône représentant un crayon. Pour modifier la plage de temps, sélectionnez l'une des valeurs prédéfinies ou choisissez custom (personnalisé). Pour plus d'informations, voir Représentation graphique d'une métrique dans le Amazon CloudWatch Guide de l'utilisateur.
7. Pour modifier les statistiques, choisissez l'onglet Graphed metrics (Graphique des métriques). Sélectionnez l'en-tête de colonne ou une valeur individuelle et choisissez une autre valeur.

Bien que vous puissiez choisir n'importe quelle statistique pour chaque métrique, toutes les statistiques ne sont pas utiles pour PredictiveScalingLoadForecast métriques. Par exemple, les calculs statistiques de moyenne, minimum et maximum de l'utilisation de l'UC sont utiles, mais le calcul statistique de somme ne l'est pas.

8. (En option) Pour ajouter une autre métrique à utiliser dans l'expression mathématique, sous Toutes les métriques, choisissez Tous, recherchez la métrique spécifique, puis activez la case à cocher en regard de celle-ci. Vous pouvez ajouter jusqu'à 10 métriques.
9. (Facultatif) Pour ajouter le graphique à un CloudWatch tableau de bord, choisissez Actions, Ajouter au tableau de bord.

Créer des mesures de précision à l'aide de mathématiques

Avec les mathématiques métriques, vous pouvez interroger plusieurs CloudWatch métriques et utilisez des expressions mathématiques pour créer de nouvelles séries chronologiques sur la base de ces métriques. Vous pouvez visualiser les séries chronologiques qui en résultent sur le CloudWatch console et ajoutez-les aux tableaux de bord. Pour plus d'informations sur les mathématiques métriques, voir Utilisation des mathématiques métriques dans le Amazon CloudWatch Guide de l'utilisateur.

À l’aide des mathématiques de métriques, vous pouvez représenter graphiquement les données générées par l’autoscaling du service pour la mise à l’échelle prédictive de différentes manières. Cela vous permet de surveiller les performances des politiques au fil du temps et de comprendre si votre combinaison de mesures peut être améliorée.

Par exemple, vous pouvez utiliser une expression mathématique de métrique pour surveillererreur absolue moyenne en pourcentage(MÂLE). La métrique MAPE permet de surveiller la différence entre les valeurs prévues et les valeurs réelles observées pendant une fenêtre de prévision donnée. Les modifications de la valeur de MAPE peuvent indiquer si les performances de la stratégie se dégradent au fil du temps à mesure que la nature de votre application change. Une augmentation de l'EMAP indique un écart plus important entre les valeurs prévues et les valeurs réelles.

Exemple : expressions mathématiques appliquées aux métriques

Pour commencer avec ce type de graphique, vous pouvez créer une expression mathématique de métrique comme celle présentée dans l'exemple suivant.

Au lieu d'une seule métrique, il existe un tableau de structures de requêtes de données de métriques pour`MetricDataQueries`. Chaque élément de`MetricDataQueries`obtient une métrique

ou exécute une expression mathématique. Le premier point,`e1`, est l'expression mathématique. L'expression désignée définit le`ReturnData`Paramètre to`true`, qui génère une seule série chronologique. Pour toutes les autres métriques, le kit`ReturnData`valeur est`false`.

Dans l'exemple, l'expression désignée utilise les valeurs réelles et prévues comme entrée et renvoie la nouvelle métrique (MAPE). `m1` est le CloudWatch métrique qui contient les valeurs de charge réelles (en supposant que l'utilisation du processeur est la métrique de charge spécifiée à l'origine pour la politique nommée `my-predictive-scaling-policy`). `m2` est le CloudWatch métrique qui contient les valeurs de charge prévues. La syntaxe mathématique de la métrique MAPE est la suivante :

Moyenne de (abs ((Réel - Forecast)/(Réel)))

Visualisez vos mesures de précision et définissez des alarmes

Pour visualiser les données de mesure de précision, sélectionnez le Métriques onglet dans CloudWatch console. Vous pouvez représenter les données sous forme graphique à partir de là. Pour plus d'informations, voir Ajouter une expression mathématique à un CloudWatch graphique dans le Amazon CloudWatch Guide de l'utilisateur.

Vous pouvez définir une alarme sur une métrique que vous surveillez à partir de la section Metrics (Métriques). Pendant que vous êtes sur l'onglet Métriques sous forme de graphique, vous pouvez sélectionner l'icône Créer une alarme sous la colonne Actions. LeCréer une alarmeest représentée par une petite cloche. Pour plus d'informations et pour connaître les options de notification, voir Création d'un CloudWatch alarme basée sur une expression mathématique métrique et Notification des utilisateurs en cas de modification des alarmes dans le Amazon CloudWatch Guide de l'utilisateur.

Vous pouvez également utiliser GetMetricData et PutMetricAlarm pour effectuer des calculs à l'aide des mathématiques métriques et créer des alarmes en fonction des résultats.

Utilisation des actions planifiées pour remplacer les valeurs prévisionnelles pour Amazon ECS

Parfois, vous pouvez disposer d'informations supplémentaires sur de futurs besoins de votre application que le calcul prédictif ne peut pas prendre en compte. Par exemple, les calculs prévisionnels pourraient sous-estimer les tâches nécessaires à la préparation d'un événement marketing à venir. Vous pouvez alors utiliser des actions planifiées pour remplacer temporairement la prévision au cours de périodes ultérieures. Les actions planifiées peuvent être exécutées de manière récurrente, ou à une date et une heure spécifiques en cas de fluctuations ponctuelles de la demande.

Par exemple, vous pouvez créer une action planifiée avec un nombre de tâches supérieur à celui prévu. Au moment de l'exécution, Amazon ECS met à jour le nombre minimum de tâches de votre service. Étant donné que la mise à l'échelle prédictive optimise le nombre de tâches, une action planifiée dont le nombre minimum de tâches est supérieur aux valeurs prévues est respectée. Cela permet d'éviter que le nombre de tâches soit inférieur à celui prévu. Pour cesser de remplacer les prévisions, utilisez une deuxième action planifiée afin de rétablir le nombre minimum de tâches à son paramètre d'origine.

La procédure suivante présente les étapes à suivre pour remplacer la prévision au cours de périodes ultérieures.

Rubriques

- Étape 1 : (facultatif) Analyser les données en séries chronologiques
- Étape 2 : créer deux actions planifiées

> **Important**
>
> Cette rubrique part du principe que vous essayez de remplacer les prévisions afin d'atteindre une capacité supérieure à celle prévue. Si vous devez réduire temporairement le nombre de tâches sans interférer avec une stratégie de mise à l'échelle prédictive, utilisez plutôt le mode prévision uniquement. En mode « prévision uniquement », la mise à l'échelle prédictive continuera de générer des prévisions, mais elle n'augmentera pas automatiquement le nombre de tâches. Vous pouvez ensuite surveiller l'utilisation des ressources et réduire manuellement le nombre de tâches selon les besoins.

Étape 1 : (facultatif) Analyser les données en séries chronologiques

Commencez par analyser les données en séries chronologiques de la prévision. Il s'agit d'une étape facultative, mais elle permet de comprendre les détails de la prévision.

1. Récupérer la prévision

   Une fois la prévision créée, vous pouvez interroger une période spécifique au sein de celle-ci. L'objectif de la requête est d'obtenir une vue complète des données en séries chronologiques d'une période spécifique.

Votre requête peut inclure jusqu'à deux jours de données de prévision ultérieures. Si vous utilisez la mise à l'échelle prédictive depuis un certain temps, vous pouvez également accéder à vos données de prévision antérieures. Toutefois, la durée maximale entre le début et la fin est de 30 jours.

Pour obtenir les prévisions à l'aide du obtenir des prévisions de mise à l'échelle prédictive AWS CLI commande, entrez les paramètres suivants dans la commande :

- Saisissez le nom du cluster dans le paramètre `resource-id`.
- Entrez le nom de la stratégie dans le paramètre `--policy-name`.
- Entrez l'heure de début dans le paramètre `--start-time` pour ne renvoyer que les données de prévision correspondant à l'heure spécifiée ou ultérieures à celle-ci.
- Entrez l'heure de fin dans le paramètre `--end-time` pour ne renvoyer que les données de prévision correspondant à l'heure spécifiée ou antérieures à celle-ci.

```
aws application-autoscaling get-predictive-scaling-forecast \
    --service-namespace ecs \
    --resource-id service/MyCluster/test \
    --policy-name cpu40-predictive-scaling-policy \
    --scalable-dimension ecs:service:DesiredCount \
    --start-time "2021-05-19T17:00:00Z" \
    --end-time "2021-05-19T23:00:00Z"
```

Si elle aboutit, la commande renvoie des données semblables à l'exemple suivant.

```
{
    "LoadForecast": [
        {
            "Timestamps": [
                "2021-05-19T17:00:00+00:00",
                "2021-05-19T18:00:00+00:00",
                "2021-05-19T19:00:00+00:00",
                "2021-05-19T20:00:00+00:00",
                "2021-05-19T21:00:00+00:00",
                "2021-05-19T22:00:00+00:00",
                "2021-05-19T23:00:00+00:00"
            ],
            "Values": [
```

```
                153.0655799339254,
                128.8288551285919,
                107.1179447150675,
                197.3601844551528,
                626.4039934516954,
                596.9441277518481,
                677.9675713779869
            ],
            "MetricSpecification": {
                "TargetValue": 40.0,
                "PredefinedMetricPairSpecification": {
                    "PredefinedMetricType": "ASGCPUUtilization"
                }
            }
        }
    ],
    "CapacityForecast": {
        "Timestamps": [
            "2021-05-19T17:00:00+00:00",
            "2021-05-19T18:00:00+00:00",
            "2021-05-19T19:00:00+00:00",
            "2021-05-19T20:00:00+00:00",
            "2021-05-19T21:00:00+00:00",
            "2021-05-19T22:00:00+00:00",
            "2021-05-19T23:00:00+00:00"
        ],
        "Values": [
            2.0,
            2.0,
            2.0,
            2.0,
            4.0,
            4.0,
            4.0
        ]
    },
    "UpdateTime": "2021-05-19T01:52:50.118000+00:00"
}
```

La réponse comprend deux prévisions : `LoadForecast` et `CapacityForecast`. `LoadForecast` affiche la prévision de charge horaire. `CapacityForecast` affiche les valeurs de prévision de la capacité nécessaire sur une base horaire pour gérer la charge prévue tout en maintenant une `TargetValue` de 40 (40 % d'utilisation moyenne du processeur).

2. Identifier la période cible

   Indiquez l'heure ou les heures où la fluctuation de la demande ponctuelle devrait avoir lieu. N'oubliez pas que les dates et les heures indiquées dans la prévision sont basées sur le fuseau horaire UTC.

Étape 2 : créer deux actions planifiées

Créez ensuite deux actions planifiées pour une période spécifique où votre application devra gérer une charge plus élevée que celle prédite. Par exemple, si vous organisez un événement marketing qui va générer du trafic sur votre site pendant une période limitée, vous pouvez planifier une action ponctuelle pour mettre à jour la capacité minimale au début de cet événement. Puis vous pouvez planifier une autre action pour rétablir le paramètre d'origine de la capacité minimale à la fin de l'événement.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, puis choisissez le service.

   La page de détails du service s’affiche.
4. Choisissez Service Auto Scaling.

   La page des stratégies s’affiche.
5. Choisissez Actions planifiées, puis Créer.

   La page Créer une action planifiée s’affiche.
6. Pour Nom de l’action, saisissez un nom unique.
7. Dans le champ Fuseau horaire, choisissez un fuseau horaire.

   Tous les fuseaux horaires répertoriés proviennent de la base de données des fuseaux horaires IANA. Pour plus d’informations, consultez la section Liste des fuseaux horaires de la base de données tz.
8. Pour Heure de début, saisissez la date et l’heure de début de l’action.
9. Pour Recurrence (Récurrence), choisissez Once (Une fois).
10. Sous Ajustements des tâches, pour « Minimum », saisissez une valeur inférieure ou égale au nombre maximum de tâches.

11. Choisissez Create Scheduled Action (Créer une action planifiée).

    La page des stratégies s'affiche.

12. Configurez une deuxième action planifiée pour rétablir le nombre minimum de tâches à la configuration d'origine à la fin de l'événement. La mise à l'échelle prédictive ne peut mettre à l'échelle le nombre de tâches que lorsque la valeur que vous avez définie pour Minimum est inférieure aux valeurs prévues.

Pour créer deux actions planifiées pour des événements ponctuels (AWS CLI)

Pour utiliser le AWS CLI pour créer les actions planifiées, utilisez le action de groupe de mise à jour planifiée commande.

Par exemple, définissons une action planifiée pour maintenir une capacité minimale de trois instances pendant huit heures à partir du 19 mai à 17h00. Les commandes suivantes montrent comment implémenter ce scénario.

La première commande put-scheduled-update-group-action indique à Amazon EC2 Auto Scaling de mettre à jour la capacité minimale du groupe Auto Scaling spécifié à 17h00 UTC le 19 mai 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-
start \
  --auto-scaling-group-name my-asg --start-time "2021-05-19T17:00:00Z" --minimum-
capacity 3
```

La deuxième commande indique à Amazon EC2 Auto Scaling de définir la capacité minimale du groupe à 1h00 UTC le 20 mai 2021.

```
aws autoscaling put-scheduled-update-group-action --scheduled-action-name my-event-end
 \
  --auto-scaling-group-name my-asg --start-time "2021-05-20T01:00:00Z" --minimum-
capacity 1
```

Après l'ajout de ces actions planifiées au groupe Auto Scaling, Amazon EC2 Auto Scaling effectue les opérations suivantes :

- À 17h00 UTC le 19 mai 2021, la première action planifiée s'exécute. Si le groupe compte actuellement moins de trois instances, il passe à trois instances. Pendant ce temps et pendant les huit heures suivantes, Amazon EC2 Auto Scaling peut continuer à monter en puissance

si la capacité prévue est supérieure à la capacité réelle ou si une stratégie de mise à l'échelle dynamique est en vigueur.

- À 1h00 UTC le 20 mai 2021, la seconde action planifiée s'exécute. Cette action rétablit le paramètre d'origine de la capacité minimale à la fin de l'événement.

Mise à l'échelle basée sur des planifications récurrentes

Pour remplacer la prévision applicable à la même période chaque semaine, créez deux actions planifiées et fournissez la logique d'heure et de date à l'aide d'une expression cron.

Le format d'expression cron se compose de cinq champs séparés par des espaces : [Minute] [Heure] [Day_of_Month] [Month_of_Year] [Day_of_Week]. Ces champs peuvent contenir toutes les valeurs autorisées, y compris des caractères spéciaux.

Par exemple, l'expression cron suivante exécute l'action tous les mardis à 6h30. L'astérisque est utilisé comme caractère générique pour représenter toutes les valeurs d'un champ.

```
30 6 * * 2
```

Consultez aussi

Pour plus d’informations sur la gestion des actions planifiées, consultez la section Utilisation des actions planifiées pour mettre à l’échelle les services Amazon ECS.

Stratégie avancée de mise à l’échelle prédictive utilisant des métriques personnalisées pour Amazon ECS

Dans une stratégie de mise à l’échelle prédictive, vous pouvez utiliser des métriques prédéfinies ou personnalisées. Les métriques personnalisées sont utiles lorsque les métriques prédéfinies (UC, mémoire, etc.) ne sont pas suffisantes pour décrire suffisamment la charge de votre application.

Lorsque vous créez une politique de dimensionnement prédictif avec des mesures personnalisées, vous pouvez spécifier d'autres CloudWatch mesures fournies par AWS. Vous pouvez également spécifier des métriques que vous définissez et publiez vous-même. Vous pouvez également utiliser les mathématiques métriques pour agréger et transformer les métriques existantes en une nouvelle série chronologique qui AWS ne suit pas automatiquement. Un exemple consiste à combiner les valeurs de vos données en calculant de nouvelles sommes ou moyennes, ce qu’on appelle l’agrégation. Les données résultantes sont appelées un agrégat.

La section suivante contient les bonnes pratiques et des exemples de construction de la structure JSON pour la politique.

Conditions préalables

Pour ajouter des métriques personnalisées à votre politique de mise à l'échelle, vous devez disposer des autorisations `cloudwatch:GetMetricData`.

Pour spécifier vos propres mesures au lieu de celles qui AWS fournit, vous devez d'abord publier vos statistiques sur CloudWatch. Pour plus d'informations, voir Publication de statistiques personnalisées dans le Amazon CloudWatch Guide de l'utilisateur.

Si vous publiez vos propres métriques, veillez à publier les points de données à une fréquence minimale de cinq minutes. Les points de données sont extraits de CloudWatch en fonction de la durée de la période dont elle a besoin. Par exemple, la spécification des métriques de charge utilise des métriques horaires pour mesurer la charge de votre application. CloudWatch utilise vos données métriques publiées pour fournir une valeur de données unique pour toute période d'une heure en agrégeant tous les points de données avec des horodatages correspondant à chaque période d'une heure.

Bonnes pratiques

Les bonnes pratiques suivantes peuvent vous aider à utiliser plus efficacement les métriques personnalisées :

- La métrique la plus utile pour la spécification de la métrique de charge est une métrique qui représente la charge sur un groupe Auto Scaling dans son ensemble.
- La métrique la plus utile pour la spécification de la métrique de mise à l’échelle est le débit moyen ou l’utilisation moyenne par tâche.
- L'utilisation cible doit correspondre au type de métrique de mise à l'échelle. Pour une configuration de stratégie qui utilise l’utilisation de l’UC, il s’agit par exemple d’un pourcentage cible.
- Si ces recommandations ne sont pas suivies, les valeurs futures prédites des séries temporelles seront probablement incorrectes. Pour vérifier que les données sont correctes, vous pouvez afficher les valeurs prévues dans la console. Sinon, après avoir créé votre politique de dimensionnement prédictif, consultez `LoadForecast` objets renvoyés par un appel au GetPredictiveScalingForecast API.
- Nous vous recommandons vivement de configurer la mise à l’échelle prédictive en mode « prévision uniquement » de manière à pouvoir évaluer les prévisions avant que la mise à l’échelle prédictive ne commence à s’appliquer activement.

Limitations

- Vous pouvez interroger des points de données de 10 métriques au maximum dans une spécification métrique.
- Dans le cadre de cette limite, une expression compte pour une métrique.

Résolution des problèmes liés à une stratégie de mise à l’échelle prédictive avec des métriques personnalisées

Si un problème survient lors de l'utilisation de métriques personnalisées, nous vous recommandons d'effectuer les opérations suivantes :

- Si vous rencontrez un problème dans un blue/green déploiement lors de l'utilisation d'une expression de recherche, assurez-vous d'avoir créé une expression de recherche qui recherche une correspondance partielle et non une correspondance exacte. Vérifiez également que la requête ne s’applique qu’aux groupes Auto Scaling exécutés dans l’application spécifique. Pour plus d'informations sur la syntaxe des expressions de recherche, voir CloudWatch syntaxe de l'expression de recherche dans le Amazon CloudWatch Guide de l'utilisateur.
- La commande put-scaling-policy valide une expression lorsque vous créez votre stratégie de mise à l’échelle. Cependant, il est possible que cette commande ne parvienne pas à identifier la cause exacte des erreurs détectées. Pour résoudre ces problèmes, dépannez les erreurs que vous recevez dans la réponse d'une requête à la commande get-metric-data. Vous pouvez également résoudre les problèmes liés à l'expression à partir du CloudWatch console.
- Vous devez spécifier `false` pour `ReturnData` si `MetricDataQueries` spécifie la fonction SEARCH() seule sans une fonction mathématique comme SUM(). Cela est dû au fait que les expressions de recherche peuvent renvoyer plusieurs séries temporelles et qu'une spécification métrique basée sur une expression ne peut renvoyer qu'une seule séries temporelles.
- Toutes les métriques impliquées dans une expression de recherche doivent avoir la même résolution.

Structure du JSON pour les métriques personnalisées de mise à l’échelle prédictive avec Amazon ECS

La section suivante contient des exemples de configuration de la mise à l'échelle prédictive pour interroger les données de CloudWatch. Il existe deux méthodes pour configurer cette option, qui affecteront le format utilisé pour créer le fichier JSON de votre politique de mise à l'échelle prédictive.

Lorsque vous utilisez des mathématiques de métriques, le format du fichier JSON varie davantage en fonction des mathématiques de métriques effectuées.

1. Pour créer une politique qui permet d'obtenir des données directement auprès d'autres CloudWatch mesures fournies par AWS ou des statistiques que vous publiez sur CloudWatch, voir Exemple de politique de dimensionnement prédictif avec des mesures de charge et de dimensionnement personnalisées à l'aide du AWS CLI.

Exemple de politique de dimensionnement prédictif avec des mesures de charge et de dimensionnement personnalisées à l'aide du AWS CLI

Pour créer une politique de dimensionnement prédictif avec des mesures de charge et de dimensionnement personnalisées à l'aide du AWS CLI, stockez les arguments pour `--predictive-scaling-configuration` dans un fichier JSON nommé `config.json`.

Vous commencez par ajouter des métriques personnalisées en remplaçant les valeurs remplaçables de l'exemple suivant par celles de vos métriques et de votre utilisation cible.

```
{
  "MetricSpecifications": [
    {
      "TargetValue": 50,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "scaling_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyUtilizationMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Average"
            }
          }
        ]
```

```
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "MyLoadMetric",
                "Namespace": "MyNameSpace",
                "Dimensions": [
                  {
                    "Name": "MyOptionalMetricDimensionName",
                    "Value": "MyOptionalMetricDimensionValue"
                  }
                ]
              },
              "Stat": "Sum"
            }
          }
        ]
      }
    }
  ]
}
```

Pour plus d'informations, voir MetricDataQuery dans le Référence de l'API Amazon EC2 Auto Scaling.

> **Note**
>
> Voici quelques ressources supplémentaires qui peuvent vous aider à trouver des noms de métriques, des espaces de noms, des dimensions et des statistiques pour CloudWatch indicateurs :
>
> - Pour plus d'informations sur les indicateurs disponibles pour AWS services, voir AWS services qui publient CloudWatch indicateurs dans le Amazon CloudWatch Guide de l'utilisateur.
> - Pour obtenir le nom exact de la métrique, l'espace de noms et les dimensions (le cas échéant) pour un CloudWatch métrique avec AWS CLI, voir métriques de liste.

Pour créer cette politique, exécutez la commande put-scaling-policy avec le fichier JSON comme entrée, tel qu'illustré dans l'exemple suivant.

```
aws application-autoscaling put-scaling-policy --policy-name my-predictive-scaling-policy \
  --auto-scaling-group-name my-asg --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

Si elle aboutit, cette commande renvoie l'Amazon Resource Name (ARN) de la stratégie.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-predictive-scaling-policy",
  "Alarms": []
}
```

Utiliser des expressions mathématiques de métrique

La section suivante fournit des informations sur l'utilisation des mathématiques de métriques avec les stratégies de mise à l'échelle prédictive dans votre stratégie.

Comprendre les mathématiques de métrique

Si vous souhaitez simplement agréger les données métriques existantes, CloudWatch les mathématiques métriques vous permettent d'économiser les efforts et les coûts liés à la publication d'une autre métrique dans CloudWatch. Vous pouvez utiliser n'importe quelle métrique qui AWS fournit, et vous pouvez également utiliser des métriques que vous définissez dans le cadre de vos applications.

Pour plus d'informations, voir Utilisation des mathématiques métriques dans le Amazon CloudWatch Guide de l'utilisateur.

Si vous choisissez d'utiliser une expression mathématique de métrique dans votre politique de mise à l'échelle prédictive, tenez compte des points suivants :

- Les opérations mathématiques métriques utilisent les points de données de la combinaison unique du nom de la métrique, de l'espace de noms et de la dimension keys/value paires de métriques.
- Vous pouvez utiliser n'importe quel opérateur arithmétique (+ - */^), fonction statistique (telle que AVG ou SUM) ou toute autre fonction qui CloudWatch supports.

- Vous pouvez utiliser à la fois des métriques et les résultats d'autres expressions mathématiques dans les formules de l'expression mathématique.
- Vos expressions mathématiques de métrique peuvent être composées de différentes agrégations. Cependant, une bonne pratique pour le résultat final de l'agrégation consiste à utiliser `Average` pour la métrique de mise à l'échelle et `Sum` pour la métrique de charge.
- Toutes les expressions utilisées dans une spécification de métrique doivent finalement retourner une seule séries temporelles.

Pour utiliser les mathématiques de métrique, procédez comme suit :

- Choisissez un ou plusieurs CloudWatch métriques. Créez ensuite l'expression. Pour plus d'informations, voir Utilisation des mathématiques métriques dans le Amazon CloudWatch Guide de l'utilisateur.
- Vérifiez que l'expression mathématique métrique est valide en utilisant CloudWatch console ou CloudWatch GetMetricData API.

Exemple de politique de dimensionnement prédictif qui combine des métriques à l'aide de mathématiques métriques (AWS CLI)

Parfois, au lieu de spécifier la métrique directement, vous devrez d'abord traiter ses données d'une certaine manière. Par exemple, une application peut extraire le travail d'une file d'attente Amazon SQS et vous souhaitez utiliser le nombre d'éléments dans la file d'attente comme critère de mise à l'échelle prédictive. Le nombre de messages dans la file d'attente ne définit pas uniquement le nombre d'instances dont vous avez besoin. Par conséquent, un travail supplémentaire est nécessaire pour créer une métrique qui peut être utilisée pour calculer le backlog par instance.

Ce qui suit est un exemple de politique de mise à l'échelle prédictive pour ce scénario. Il spécifie les métriques de mise à l'échelle et de charge qui sont basées sur la métrique `ApproximateNumberOfMessagesVisible` d'Amazon SQS, qui est le nombre de messages disponibles pour la récupération de la file d'attente. Il utilise également la métrique `GroupInServiceInstances` d'Amazon EC2 Auto Scaling et une expression mathématique pour calculer le backlog par instance pour la métrique de mise à l'échelle.

```
aws application-autoscaling put-scaling-policy --policy-name my-sqs-custom-metrics-
policy \
  --policy-type PredictiveScaling \
  --predictive-scaling-configuration file://config.json
```

```
  --service-namespace ecs \
  --resource-id service/MyCluster/test \
  "MetricSpecifications": [
    {
      "TargetValue": 100,
      "CustomizedScalingMetricSpecification": {
        "MetricDataQueries": [
          {
            "Label": "Get the queue size (the number of messages waiting to be
processed)",
            "Id": "queue_size",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
          {
            "Label": "Get the group size (the number of running instances)",
            "Id": "running_capacity",
            "MetricStat": {
              "Metric": {
                "MetricName": "GroupInServiceInstances",
                "Namespace": "AWS/AutoScaling",
                "Dimensions": [
                  {
                    "Name": "AutoScalingGroupName",
                    "Value": "my-asg"
                  }
                ]
              },
              "Stat": "Sum"
            },
            "ReturnData": false
          },
```

```
          {
            "Label": "Calculate the backlog per instance",
            "Id": "scaling_metric",
            "Expression": "queue_size / running_capacity",
            "ReturnData": true
          }
        ]
      },
      "CustomizedLoadMetricSpecification": {
        "MetricDataQueries": [
          {
            "Id": "load_metric",
            "MetricStat": {
              "Metric": {
                "MetricName": "ApproximateNumberOfMessagesVisible",
                "Namespace": "AWS/SQS",
                "Dimensions": [
                  {
                    "Name": "QueueName",
                    "Value": "my-queue"
                  }
                ],
              },
              "Stat": "Sum"
            },
            "ReturnData": true
          }
        ]
      }
    }
  ]
}
```

L'exemple renvoie l'ARN de la politique.

```
{
  "PolicyARN": "arn:aws:autoscaling:region:account-id:scalingPolicy:2f4f5048-d8a8-4d14-
b13a-d1905620f345:autoScalingGroupName/my-asg:policyName/my-sqs-custom-metrics-policy",
  "Alarms": []
}
```

Faites évoluer les services Amazon ECS en fonction d'une file d'attente Amazon SQS

Vous pouvez adapter votre service Amazon ECS en fonction de la charge d'une file d'attente Amazon Simple Queue Service (Amazon SQS). Cela est utile pour les applications asynchrones dans lesquelles les tâches de votre service agissent en tant que travailleurs qui consomment et traitent les messages d'une file d'attente. À mesure que le taux de messages entrants change, Amazon ECS Service Auto Scaling ajoute des tâches lorsque la file d'attente s'accumule et supprime des tâches lorsqu'elle est épuisée. Cela permet de maintenir la latence des messages en attente dans la file d'attente à un niveau acceptable sans surprovisionnement.

Utiliser la métrique du backlog par tâche

Amazon SQS publie la profondeur d'une file d'attente pour CloudWatch en tant que `ApproximateNumberOfMessagesVisible` métrique, qui indique le nombre de messages disponibles pour la récupération. La profondeur de la file d'attente indique qu'il y a du travail à faire, mais il ne s'agit pas d'une métrique fiable sur laquelle il est possible de s'appuyer directement. Le nombre de messages dans la file d'attente ne change pas proportionnellement au nombre de tâches qui la traitent. Le nombre de tâches dont vous avez besoin dépend également d'autres facteurs, tels que le temps nécessaire pour traiter un message et le niveau de latence acceptable (délai de file d'attente). Une politique de suivi des cibles qui suit la profondeur de la file d'attente interprète une file d'attente importante comme une demande pour un plus grand nombre de tâches. Cela peut entraîner une extension de votre service bien au-delà de la charge de travail requise et une augmentation des coûts.

Au lieu de cela, calculez sur arriéré par tâche: la profondeur de la file d'attente divisée par le nombre de tâches en cours d'exécution. Vous calculez cette valeur à l'aide d'un CloudWatch expression mathématique métrique qui divise la profondeur de la file d'attente par le nombre de tâches en cours. Le backlog par tâche normalise la profondeur de la file d'attente en fonction du nombre de tâches en cours. Cela le rend proportionnel à la charge de chaque tâche et convient donc à la mise à l'échelle.

Définissez votre valeur cible sur arriéré acceptable par tâche. Pour le calculer, divisez la latence que votre application peut accepter par le temps moyen nécessaire à une tâche pour traiter un message.

Par exemple, si la latence acceptable est de 10 secondes et qu'une tâche prend 0,1 seconde pour traiter un message, le backlog acceptable par tâche est de 10 divisé par 0,1, ce qui équivaut à 100 messages par tâche. Il s'agit de la valeur cible de votre police. Si la file d'attente contient actuellement 1 500 messages et que 10 tâches sont en cours d'exécution, le backlog par tâche est de 150 (1 500 divisés par 10), ce qui dépasse l'objectif. Votre service est divisé en cinq tâches afin de ramener l'arriéré par tâche au prorata de la valeur cible.

Utiliser les mathématiques métriques pour calculer la métrique

Prise en charge des politiques de suivi des cibles Application Auto Scaling CloudWatch mathématiques métriques. Les mathématiques métriques vous permettent d'interroger plusieurs CloudWatch métriques et utilisez des expressions mathématiques pour créer une nouvelle série chronologique basée sur ces métriques. Vous utilisez une expression mathématique métrique pour calculer le backlog par tâche directement dans la politique de dimensionnement. L'application Auto Scaling évalue l'expression et gère CloudWatch des alarmes qui déclenchent les activités de dimensionnement. Pour plus d'informations, voir Utiliser les mathématiques métriques dans le CloudWatch Guide de l'utilisateur.

Pour calculer le backlog par tâche, vous utilisez les deux méthodes suivantes CloudWatch indicateurs :

- `ApproximateNumberOfMessagesVisible` (les `Sum` statistique) à partir du `AWS/SQS` espace de noms, pour le nombre de messages en attente dans la file d'attente.
- `RunningTaskCount` (les `Average` statistique) à partir du `ECS/ContainerInsights` espace de noms, pour le nombre de tâches en cours d'exécution dans votre service. Pour utiliser cette métrique, vous devez activer CloudWatch Container Insights pour votre cluster. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.

Dans l'exemple suivant, `m1` est la profondeur de la file d'attente et `m2` est le nombre de tâches en cours. L'expression mathématique métrique `m1 / m2` produit le backlog par tâche. Application Auto Scaling ajuste le nombre de tâches de votre service pour maintenir cette valeur à votre valeur cible.

Utilisez la mise à l'échelle par étapes pour plus de contrôle

Nous recommandons le suivi des cibles pour la plupart des charges de travail basées sur des files d'attente, car il s'agit de l'approche la plus simple. Vous définissez une valeur cible unique, et Amazon ECS Service Auto Scaling crée et gère CloudWatch les alarmes et les réglages de mise à l'échelle pour vous.

Utilisez la mise à l'échelle par étapes lorsque vous souhaitez contrôler totalement le comportement de mise à l'échelle. Grâce à la mise à l'échelle par étapes, vous créez votre propre CloudWatch alarmes et définissez des politiques de scale-out et de scale-in distinctes, chacune avec ses propres ajustements d'étape. Vous insérez l'expression mathématique de la métrique du backlog par tâche dans les alarmes et définissez l'action de chaque alarme selon la politique correspondante. Cela vous permet de contrôler les éléments suivants :

- La taille de chaque étape de mise à l'échelle — Vous pouvez définir des ajustements par étapes qui ajoutent d'autres tâches à mesure que l'arriéré augmente, de sorte qu'une petite faille ajoute quelques tâches et une violation importante en ajoute plusieurs. Vous pouvez effectuer une mise à l'échelle en fonction d'un pourcentage du nombre de tâches en cours ou d'un nombre fixe de tâches.
- La rapidité avec laquelle le dimensionnement réagit — Vous contrôlez la réactivité grâce à la fréquence de publication des métriques et aux paramètres d'évaluation des alarmes. Par exemple, vous pouvez publier une métrique de file d'attente en haute résolution toutes les 10 secondes et configurer une alarme de scale-out pour contourner un point de données unique, afin que le service réagisse à un arriéré croissant plus rapidement que ne le permet la résolution par défaut d'une minute.
- Comportement différent lors de la mise à l'échelle et de la mise à l'échelle — Étant donné que le scale-out et le scale-in sont des politiques et des alarmes distinctes, vous pouvez réagir rapidement à un arriéré croissant tout en procédant à une mise à l'échelle de manière prudente. Par exemple, vous pouvez exiger que l'arriéré reste faible pendant plusieurs minutes avant de supprimer des tâches.

Pour plus d’informations, consultez Utilisez des incréments prédéfinis en fonction de CloudWatch alarmes pour faire évoluer les services Amazon ECS.

Protégez les tâches qui traitent des messages

Lorsque votre service évolue, Amazon ECS Service Auto Scaling peut arrêter une tâche qui traite toujours un message. Dans le cas d'un travail de longue durée ou non idempotent, cela peut entraîner le retraitement du message ou la perte de sa progression. Pour éviter cela, utilisez la protection évolutive des tâches pour marquer une tâche comme protégée pendant qu'elle est en cours de travail.

Pour les personnes chargées des files d'attente, nous recommandons que la tâche se protège elle-même via le point de terminaison de l'agent de conteneur Amazon ECS. Lorsque la tâche commence à traiter un message, elle définit le `ProtectionEnabled` attribuer à `true` afin qu'Amazon ECS ne l'arrête pas lors d'un événement de scale-in. Une fois que la tâche a terminé le message, elle définit `ProtectionEnabled` pour `false` afin que la tâche puisse à nouveau être résiliée. Pour plus d’informations, consultez Protection de vos tâches Amazon ECS pour éviter qu’elles ne soient interrompues par des événements de réduction horizontale.

Considérations

Lorsque vous configurez la mise à l'échelle basée sur les files d'attente pour votre service Amazon ECS, tenez compte des points suivants :

- Étant donné que Application Auto Scaling évalue l'expression mathématique des métriques pour vous, vous n'avez pas besoin de publier de métrique personnalisée ni d'exécuter de code supplémentaire pour calculer le backlog par tâche.
- Le `RunningTaskCount` la métrique n'est disponible que lorsque CloudWatch Container Insights est activé pour votre cluster. Si ce n'est pas le cas, l'expression mathématique métrique ne contient aucune donnée pour le diviseur et la mise à l'échelle n'a pas lieu.
- Choisissez une valeur cible en fonction de la latence autorisée par le contrat de niveau de service (SLA) de votre application. Une valeur cible inférieure réduit le délai de file d'attente tout en exécutant plus de tâches. Une valeur cible plus élevée réduit les coûts mais augmente le temps d'attente des messages dans la file d'attente.
- Les messages dont le traitement n'est pas terminé lorsqu'une tâche est arrêtée sont renvoyés dans la file d'attente Amazon SQS. Une autre tâche en cours peut les traiter si vous configurez correctement le délai de visibilité. Pour empêcher l'arrêt d'une tâche pendant le traitement d'un message, utilisez la protection évolutive des tâches, comme décrit dans Protégez les tâches qui traitent des messages.
- Les considérations générales relatives à Amazon ECS Service Auto Scaling et au suivi des cibles s'appliquent également à la mise à l'échelle basée sur les files d'attente.

Configurer le dimensionnement en fonction des files d'attente

Les étapes suivantes montrent comment créer une politique de dimensionnement du suivi des cibles qui utilise une expression mathématique métrique pour dimensionner votre service Amazon ECS en fonction du backlog par tâche. Vous utilisez le AWS CLI pour ces tâches.

1. Vérifiez que vous disposez des autorisations IAM requises pour créer et mettre à jour des services et pour enregistrer des cibles évolutives et des politiques de dimensionnement. Pour plus d’informations, consultez Autorisations IAM requises pour l’autoscaling d’un service Amazon ECS.
2. Allumer CloudWatch Container Insights pour votre cluster afin que `RunningTaskCount` la métrique est disponible. Pour plus d’informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.

3. Enregistrez votre service Amazon ECS en tant que cible évolutive à l'aide du register-scalable-target commande. La dimension évolutive est `ecs:service:DesiredCount`.

```
aws application-autoscaling register-scalable-target \
    --service-namespace ecs \
    --scalable-dimension ecs:service:DesiredCount \
    --resource-id service/my-cluster/my-service \
    --min-capacity 1 \
    --max-capacity 10
```

4. Stockez l'expression mathématique métrique dans une spécification métrique personnalisée dans un fichier JSON nommé `config.json`. Remplacez le nom de la file d'attente, le nom du cluster et le nom du service par vos propres valeurs, puis définissez `TargetValue` au backlog acceptable par tâche que vous avez calculé.

```
{
    "CustomizedMetricSpecification": {
        "Metrics": [
            {
                "Label": "Get the queue depth",
                "Id": "m1",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "ApproximateNumberOfMessagesVisible",
                        "Namespace": "AWS/SQS",
                        "Dimensions": [
                            { "Name": "QueueName", "Value": "my-queue" }
                        ]
                    },
                    "Stat": "Sum"
                },
                "ReturnData": false
            },
            {
                "Label": "Get the running task count",
                "Id": "m2",
                "MetricStat": {
                    "Metric": {
                        "MetricName": "RunningTaskCount",
                        "Namespace": "ECS/ContainerInsights",
                        "Dimensions": [
                            { "Name": "ClusterName", "Value": "my-cluster" },
```

```
                            { "Name": "ServiceName", "Value": "my-service" }
                        ]
                    },
                    "Stat": "Average"
                },
                "ReturnData": false
            },
            {
                "Label": "Calculate the backlog per task",
                "Id": "e1",
                "Expression": "m1 / m2",
                "ReturnData": true
            }
        ]
    },
    "TargetValue": 100
}
```

5. Créez la politique de dimensionnement du suivi des cibles à l'aide du politique de mise à l'échelle commande avec le fichier JSON en entrée. L'application Auto Scaling crée et gère les deux CloudWatch alarmes (une pour le scale-out et l'autre pour le scale-in) qui invoquent la politique.

```
aws application-autoscaling put-scaling-policy \
    --policy-name sqs-backlog-target-tracking-policy \
    --service-namespace ecs \
    --resource-id service/my-cluster/my-service \
    --scalable-dimension ecs:service:DesiredCount \
    --policy-type TargetTrackingScaling \
    --target-tracking-scaling-policy-configuration file://config.json
```

Pour de plus amples informations sur les politiques de dimensionnement de suivi de cible, veuillez consulter Politiques de dimensionnement de suivi de cible dans le Manuel de l'utilisateur Application Auto Scaling.

Interconnexion des services Amazon ECS

Les applications qui s'exécutent dans le cadre de tâches Amazon ECS ont souvent besoin de recevoir des connexions depuis Internet ou de se connecter à d'autres applications exécutées dans les services Amazon ECS. Si vous avez besoin de connexions externes depuis Internet, nous vous recommandons d'utiliser Elastic Load Balancing. Pour plus d'informations sur l'équilibrage de charge

intégrée, consultez the section called “Utilisation de l’équilibrage de charge pour répartir le trafic des services”.

Si vous avez besoin d'une application pour vous connecter à d'autres applications qui s'exécutent dans les services Amazon ECS, Amazon ECS propose les méthodes suivantes pour le faire sans équilibreur de charge :

- Amazon ECS Service Connect

  Nous recommandons Service Connect, qui fournit une configuration Amazon ECS pour la découverte de service, la connectivité et la surveillance du trafic. Avec Service Connect, vos applications peuvent utiliser des noms abrégés et des ports standard pour se connecter aux services Amazon ECS du même cluster ou à d’autres clusters, y compris entre les VPC de la même Région AWS.

  Lorsque vous utilisez Service Connect, Amazon ECS gère toutes les étapes de la découverte de services : création des noms susceptibles d’être découverts, gestion dynamique des entrées pour chaque tâche au démarrage et à l’arrêt des tâches, exécution d’un agent dans chaque tâche configuré pour découvrir les noms. Votre application peut rechercher les noms en utilisant les fonctionnalités standard pour les noms DNS et en établissant des connexions. Si votre application le fait déjà, vous n’avez pas besoin de modifier votre application pour utiliser Service Connect.

  Vous fournissez la configuration complète dans chaque définition de service et de tâche. Amazon ECS gère les modifications apportées à cette configuration lors de chaque déploiement de service, afin de garantir que toutes les tâches d’un déploiement se comportent de la même manière. Par exemple, un problème courant avec le DNS en tant que découverte de service est le contrôle d’une migration. Si vous modifiez un nom DNS pour qu’il pointe vers les nouvelles adresses IP de remplacement, le délai TTL maximal peut être nécessaire avant que tous les clients commencent à utiliser le nouveau service. Avec Service Connect, le déploiement du client met à jour la configuration en remplaçant les tâches du client. Vous pouvez configurer le disjoncteur de déploiement et toute autre configuration de déploiement pour affecter les modifications de Service Connect de la même manière que pour tout autre déploiement.

  Pour plus d’informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.
- Découverte du service Amazon ECS

  Une autre approche de la communication de service à service est la communication directe utilisant la découverte de service. Dans cette approche, vous pouvez utiliser l’intégration de la découverte

de service AWS Cloud Map avec Amazon ECS. À l'aide de la découverte de services, Amazon ECS synchronise la liste des tâches lancées avec AWS Cloud Map, qui gère un nom d'hôte DNS qui renvoie aux adresses IP internes d'une ou de plusieurs tâches de ce service particulier. D'autres services au sein d'Amazon VPC peuvent utiliser ce nom d'hôte DNS pour envoyer le trafic directement vers un autre conteneur en utilisant son adresse IP interne.

Cette approche de communication de service à service fournit une faible latence. Il n'y a aucun composant supplémentaire entre les conteneurs. Le trafic passe directement d'un conteneur à l'autre.

Cette approche convient lorsque vous utilisez le mode réseau `awsvpc`, où chaque tâche possède sa propre adresse IP unique. La plupart des logiciels ne prennent en charge que l'utilisation d'enregistrements DNS `A`, qui se résolvent directement en adresses IP. Lorsque vous utilisez le mode réseau `awsvpc`, l'adresse IP de chaque tâche est un enregistrement `A`. Toutefois, si vous utilisez le mode réseau `bridge`, plusieurs conteneurs peuvent partager la même adresse IP. En outre, les mappages de ports dynamiques entraînent l'attribution aléatoire de numéros de port aux conteneurs sur cette adresse IP unique. À ce stade, un enregistrement `A` ne suffit plus pour la découverte de service. Vous devez également utiliser un enregistrement `SRV`. Ce type d'enregistrement permet de suivre à la fois les adresses IP et les numéros de port, mais nécessite que vous configuriez les applications de manière appropriée. Certaines applications prédéfinies que vous utilisez peuvent ne pas prendre en charge les enregistrements `SRV`.

Un autre avantage du mode réseau `awsvpc` est que vous disposez d'un groupe de sécurité unique pour chaque service. Vous pouvez configurer ce groupe de sécurité pour autoriser les connexions entrantes provenant uniquement des services en amont spécifiques qui doivent communiquer avec ce service.

Le principal inconvénient de la communication directe entre services utilisant la découverte de service est que vous devez implémenter une logique supplémentaire pour effectuer de nouvelles tentatives et faire face aux échecs de connexion. Les enregistrements DNS ont une période de durée de vie (TTL) qui détermine combien de temps ils sont mis en cache. Il faut un certain temps pour que l'enregistrement DNS soit mis à jour et que le cache expire afin que vos applications puissent récupérer la dernière version de l'enregistrement DNS. Il se peut donc que votre application finisse par résoudre l'enregistrement DNS pour qu'il pointe vers un autre conteneur qui n'existe plus. Votre application doit gérer les nouvelles tentatives et avoir une logique lui permettant d'ignorer les mauvais dorsaux.

Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS

- Amazon VPC Lattice

  Amazon VPC Lattice est un service de réseau d'applications géré que les clients d'Amazon ECS utilisent pour observer, sécuriser et surveiller les applications créées sur AWS services informatiques, VPC et comptes sans avoir à modifier leur code.

  VPC Lattice utilise des groupes cibles, qui sont un ensemble de ressources de calcul. Ces cibles exécutent votre application ou votre service et peuvent être des instances Amazon EC2, des adresses IP, des fonctions Lambda et des équilibreurs de charge d'application. En associant leurs services Amazon ECS à un groupe cible VPC Lattice, les clients peuvent désormais activer les tâches Amazon ECS en tant que cibles IP dans VPC Lattice. Amazon ECS enregistre automatiquement les tâches auprès du groupe cible VPC Lattice lorsque les tâches du service enregistré sont lancées.

  Pour plus d'informations, consultez Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS.

Tableau de compatibilité des modes réseau

Le tableau suivant décrit la compatibilité entre ces options et les modes réseau des tâches. Dans le tableau, le terme « client » fait référence à l'application qui établit les connexions depuis une tâche Amazon ECS.

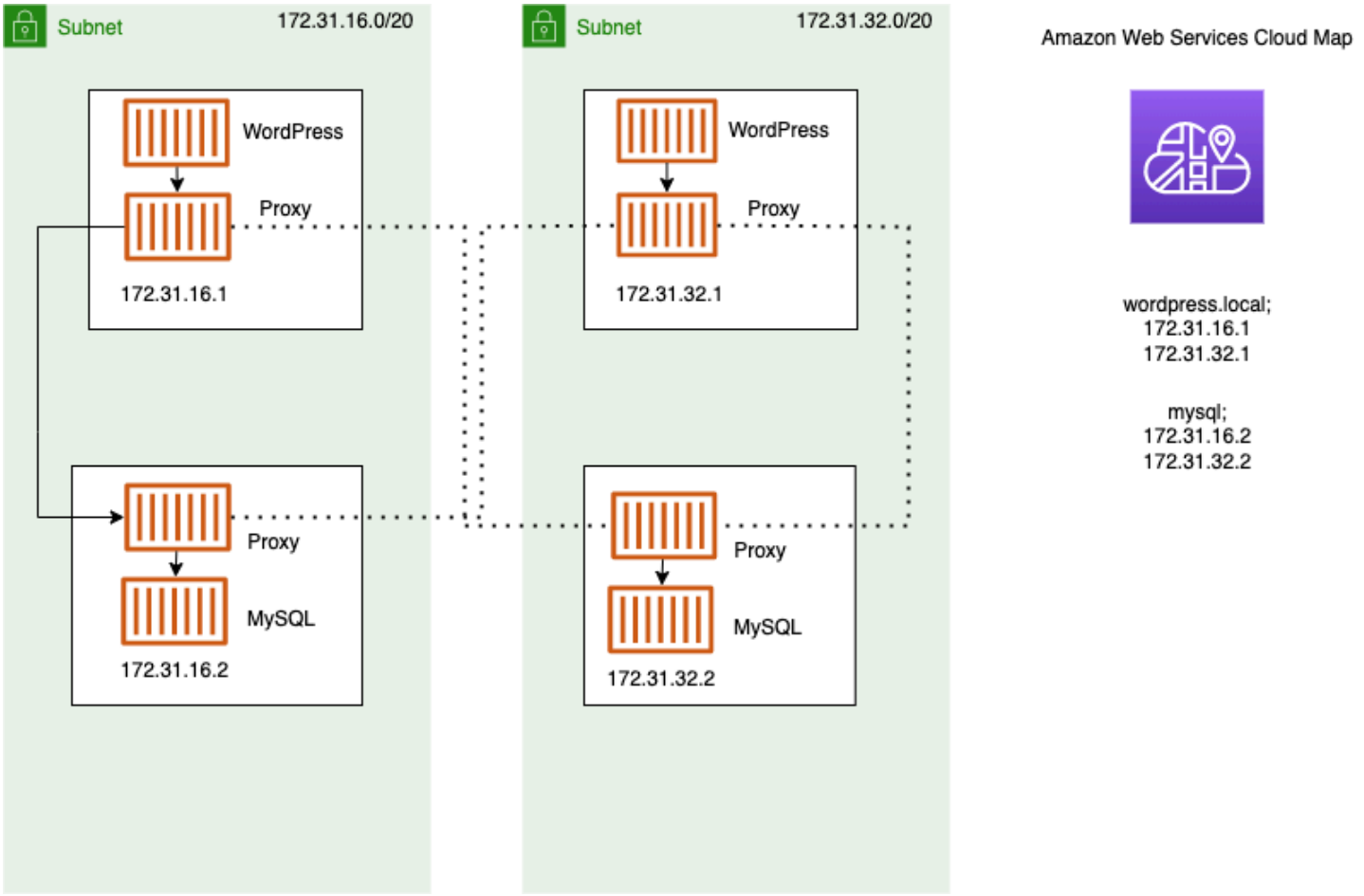
Options d'interconnexion	Pontée	**`awsvpc`**	Host (Hôte)
Découverte de service	oui, mais nécessite que les clients soient au courant des enregistr ements SRV dans le DNS sans `hostPort`.	oui	oui, mais nécessite que les clients soient au courant des enregistr ements SRV dans le DNS sans `hostPort`.
Service Connect	oui	oui	non

Options d'interconnexion	Pontée	**awsvpc**	Host (Hôte)
Treillis en PVC	oui	oui	oui

Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés

Amazon ECS Service Connect permet de gérer les communications de service à service en tant que configuration Amazon ECS. Il crée à la fois la découverte de services et un maillage de services dans Amazon ECS. Il fournit la configuration complète de chaque service que vous gérez par le biais de déploiements de services, une manière unifiée de faire référence à vos services dans des espaces de noms qui ne dépendent pas de la configuration DNS du VPC, ainsi que des métriques et des journaux normalisés pour surveiller toutes vos applications. Service Connect n'interconnecte que les services.

Le schéma suivant montre un exemple de réseau Service Connect avec deux sous-réseaux dans le VPC et deux services. Un service client qui fonctionne WordPress avec 1 tâche dans chaque sous-réseau. Un service de serveur qui exécute MySQL avec une tâche dans chaque sous-réseau. Les deux services sont hautement disponibles et résilients aux problèmes liés aux tâches et aux zones de disponibilité, car chaque service exécute plusieurs tâches réparties sur deux sous-réseaux. Les flèches pleines indiquent une connexion entre WordPress à MySQL. Par exemple, un `mysql --host=mysql` Commande CLI exécutée depuis l'intérieur du WordPress conteneur dans la tâche avec l'adresse IP `172.31.16.1`. La commande utilise le nom abrégé `mysql` du port par défaut de MySQL. Ce nom et ce port se connectent au proxy Service Connect dans le cadre de la même tâche. Le proxy dans WordPress task utilise un équilibrage de charge circulaire et toutes les informations d'échec précédentes lors de la détection des valeurs aberrantes pour sélectionner la tâche MySQL à laquelle se connecter. Comme le montrent les flèches pleines du schéma, le proxy se connecte au deuxième proxy de la tâche MySQL avec l'adresse IP `172.31.16.2`. Le second proxy se connecte au serveur MySQL local dans le cadre de la même tâche. Les deux proxys indiquent des performances de connexion qui sont visibles dans les graphiques d'Amazon ECS et d'Amazon CloudWatch consoles afin que vous puissiez obtenir des mesures de performance de toutes sortes d'applications de la même manière.



Les services suivants sont utilisés avec Service Connect.

nom du port

Configuration de définition de tâche Amazon ECS qui attribue un nom à un mappage de port particulier. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

alias du client

Configuration du service Amazon ECS qui attribue le numéro de port utilisé dans le point de terminaison. L'alias du client peut également attribuer le nom DNS du point de terminaison, en remplaçant le nom de découverte. Si aucun nom de découverte n'est fourni dans le service Amazon ECS, le nom d'alias du client remplace le nom du port comme nom de point de terminaison. Pour des exemples de points de terminaison, consultez la définition de point de terminaison. Plusieurs alias client peuvent être attribués à un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

nom de découverte

Nom intermédiaire facultatif que vous pouvez créer pour un port spécifié à partir de la définition de tâche. Ce nom est utilisé pour créer un AWS Cloud Map service. Si ce nom n'est pas fourni, le nom de port indiqué dans la définition de tâche est utilisé. Plusieurs noms de découverte peuvent être attribués à un port spécifique d'un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

AWS Cloud Map les noms de service doivent être uniques au sein d'un espace de noms. En raison de cette limitation, vous ne pouvez avoir qu'une seule configuration Service Connect sans nom de découverte pour une définition de tâche particulière dans chaque espace de noms.

point de terminaison

URL permettant de se connecter à une API ou à un site Web. L'URL contient le protocole, un nom DNS et le port. Pour plus d'informations sur les points de terminaison en général, veuillez consulter point de terminaison dans le glossaire AWS de la Référence générale d'Amazon Web Services(langue française non garantie).

Service Connect crée des points de terminaison qui se connectent aux services Amazon ECS et configure les tâches des services Amazon ECS pour se connecter aux points de terminaison. L'URL contient le protocole, un nom DNS et le port. Vous sélectionnez le protocole et le nom du port dans la définition de la tâche, car le port doit correspondre à l'application qui se trouve dans l'image du conteneur. Dans le service, vous sélectionnez chaque port par son nom et vous pouvez attribuer le nom DNS. Si vous ne spécifiez pas de nom DNS dans la configuration du service Amazon ECS, le nom de port indiqué dans la définition de tâche est utilisé par défaut. Par exemple, un point de terminaison Service Connect pourrait être `http://blog:80`, `grpc://checkout:8080` ou `http://_db.production.internal:99`.

Service Service Connect

Configuration d'un point de terminaison unique dans un service Amazon ECS. Cela fait partie de la configuration de Service Connect, qui se compose d'une seule ligne dans la configuration de Service Connect et du nom de découverte dans la console, ou d'un objet dans la liste `services` de la configuration JSON d'un service Amazon ECS. Cette configuration est uniquement utilisée par Amazon ECS Service Connect.

Pour plus d'informations, voir ServiceConnectService dans la référence de l'API Amazon Elastic Container Service.

espace de nom

Le nom abrégé ou le nom de ressource Amazon complet (ARN) du AWS Cloud Map espace de noms à utiliser avec Service Connect. L'espace de noms doit être dans le même Région AWS en tant que service et cluster Amazon ECS. Le type d'espace de noms dans AWS Cloud Map n'affecte pas Service Connect. L'espace de noms peut être partagé avec votre Compte AWS en utilisant AWS Resource Access Manager (AWS RAM) dans Régions AWS cette AWS RAM est disponible en. Pour plus d'informations sur les espaces de noms partagés, consultez Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

Service Connect utilise AWS Cloud Map espace de noms en tant que regroupement logique de tâches Amazon ECS qui communiquent entre elles. Chaque service Amazon ECS ne peut appartenir qu'à un seul espace de noms. Les services d'un espace de noms peuvent être répartis sur différents clusters Amazon ECS au sein d'une même Région AWS. Si l'espace de noms est un espace de noms partagé, les services peuvent être répartis entre le propriétaire et le consommateur de l'espace de noms Comptes AWS. Vous pouvez organiser librement les services selon n'importe quel critère.

service client

Un service Amazon ECS qui exécute une application client réseau. Un espace de noms doit être configuré pour ce service. Chaque tâche du service peut découvrir et se connecter à tous les points de terminaison de l'espace de noms via un conteneur de proxy Service Connect.

Si l'un des conteneurs de la tâche doit se connecter à un point de terminaison à partir d'un service situé dans un espace de noms, choisissez un service client. Si une application frontend, un proxy inverse ou une application d'équilibreur de charge reçoit du trafic externe via d'autres méthodes comme Elastic Load Balancing, elle peut utiliser ce type de configuration Service Connect.

service client-serveur

Service Amazon ECS qui exécute une application service réseau ou web. Un espace de noms et au moins un point de terminaison doivent être configurés pour ce service. Chaque tâche du service est accessible à l'aide des points de terminaison. Le conteneur de proxy Service Connect écoute le nom et le port du point de terminaison pour diriger le trafic vers les conteneurs d'applications de la tâche.

Si l'un des conteneurs expose et écoute le trafic réseau sur un port, choisissez un service client-serveur. Ces applications n'ont pas besoin de se connecter à d'autres services client-serveur dans

le même espace de noms, mais la configuration du client est nécessaire. Un dorsal, un intergiciel, une couche entreprise ou la plupart des microservices utilisent ce type de configuration Service Connect. Si vous souhaitez qu'une application frontend, un proxy inverse ou une application d'équilibreur de charge reçoive du trafic provenant d'autres services configurés avec Service Connect dans le même espace de noms, ces services doivent utiliser ce type de configuration Service Connect.

La fonctionnalité Service Connect crée un réseau virtuel de services connexes. La même configuration de service peut être utilisée sur plusieurs espaces de noms différents pour exécuter des ensembles d'applications indépendants mais identiques. Service Connect définit le conteneur de proxy dans le service Amazon ECS. Ainsi, la même définition de tâche peut être utilisée pour exécuter des applications identiques dans différents espaces de noms avec des configurations Service Connect différentes. Chaque tâche réalisée par le service exécute un conteneur de proxy dans la tâche.

Service Connect convient aux connexions entre les services Amazon ECS au sein du même espace de noms. Pour les applications suivantes, vous devez utiliser une méthode d'interconnexion supplémentaire pour vous connecter à un service Amazon ECS configuré avec Service Connect :

- Tâches configurées dans d’autres espaces de noms
- Tâches qui ne sont pas configurées pour Service Connect
- Autres applications en dehors d’Amazon ECS

Ces applications peuvent se connecter via le proxy Service Connect mais ne peuvent pas résoudre les noms des points de terminaison Service Connect.

Pour que ces applications puissent résoudre les adresses IP des tâches Amazon ECS, vous devez utiliser une autre méthode d’interconnexion.

Rubriques

- Tarification
- Composants d’Amazon ECS Service Connect
- Vue d’ensemble de la configuration d’Amazon ECS Service Connect
- Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms
- Journaux d'accès à Amazon ECS Service Connect
- Routage tenant compte des zones Amazon ECS Service Connect

- Chiffrement du trafic Amazon ECS Service Connect
- Configuration du conteneur proxy Service Connect
- Configuration d'Amazon ECS Service Connect à l'aide du AWS CLI

Tarification

- La tarification d'Amazon ECS Service Connect varie selon que vous utilisez AWS Fargate ou une infrastructure Amazon EC2 pour héberger vos charges de travail conteneurisées. Lorsque vous utilisez Amazon ECS sur AWS Outposts, la tarification suit le même modèle que lorsque vous utilisez directement Amazon EC2. Pour plus d'informations, consultez Tarification Amazon ECS.
- L’utilisation d’Amazon ECS Service Connect n’entraîne pas de frais supplémentaires.
- Il n'y a pas de frais supplémentaires pour AWS Cloud Map utilisation lorsqu'elle est utilisée par Service Connect.
- Les clients paient pour les ressources de calcul utilisées par Amazon ECS Service Connect, notamment le vCPU et la mémoire. Étant donné que l’agent Amazon ECS Service Connect s’exécute dans une tâche client, son utilisation n’entraîne pas de frais supplémentaires. Les ressources de tâche sont partagées entre la charge de travail du client et l’agent Amazon ECS Service Connect.
- Lorsque vous utilisez la fonctionnalité de chiffrement du trafic Amazon ECS Service Connect avec CA privée AWS, les clients paient pour l'autorité de certification privée qu'ils créent et pour chaque certificat TLS émis. Pour plus d’informations, consultez la section Tarification Autorité de certification privée AWS. Pour estimer le coût mensuel des certificats TLS, les clients doivent connaître le nombre de services Amazon ECS sur lesquels le protocole TLS est activé, le multiplier par le coût du certificat, puis le multiplier par six. Amazon ECS Service Connect faisant automatiquement alterner les certificats TLS tous les cinq jours, six certificats sont émis par service Amazon ECS, par mois, en moyenne.

Composants d’Amazon ECS Service Connect

Lorsque vous utilisez Amazon ECS Service Connect, vous configurez chaque service Amazon ECS pour exécuter une application serveur qui reçoit des requêtes réseau (service client-serveur) ou pour exécuter une application cliente qui fait les requêtes (service client).

Lorsque vous vous préparez à utiliser Service Connect, commencez par un service client-serveur. Vous pouvez ajouter une configuration Service Connect à un nouveau service ou à un service existant. Amazon ECS crée un point de terminaison Service Connect dans l’espace de noms. En

outre, Amazon ECS crée un nouveau déploiement dans le service pour remplacer les tâches en cours d'exécution.

Les tâches existantes et les autres applications peuvent continuer à se connecter aux points de terminaison existants et aux applications externes. Si un service client-serveur ajoute des tâches par augmentation horizontale, les nouvelles connexions provenant des clients seront réparties de manière équilibrée entre toutes les tâches. Si un service client-serveur est mis à jour, les nouvelles connexions provenant des clients seront réparties de manière équilibrée entre les tâches de la nouvelle version.

Les tâches existantes ne peuvent pas être résolues et se connecter au nouveau point de terminaison. Seules les nouvelles tâches disposant d'une configuration Service Connect dans le même espace de noms et qui commencent à s'exécuter après ce déploiement peuvent résoudre et se connecter à ce point de terminaison.

Cela signifie que l'opérateur de l'application cliente détermine quand la configuration de son application change, même si l'opérateur de l'application serveur peut modifier sa configuration à tout moment. La liste des points de terminaison dans l'espace de noms peut changer à chaque fois qu'un service de cet espace de noms est déployé. Les tâches existantes et les tâches de remplacement continuent de se comporter comme elles le faisaient après le déploiement le plus récent.

Considérez les exemples suivants.

Tout d'abord, supposons que vous créez une application disponible sur Internet public en un seul AWS CloudFormation modèle et unique CloudFormation pile. La découverte et l'accessibilité publiques devraient être créées en dernier lieu CloudFormation, y compris le service client frontal. Les services doivent être créés dans cet ordre afin d'éviter toute période pendant laquelle le service client frontend fonctionne et est accessible au public, alors que le backend ne l'est pas. Cela permet d'éliminer l'envoi de messages d'erreur au public pendant cette période. Dans AWS CloudFormation, vous devez utiliser `dependsOn` pour indiquer à CloudFormation que plusieurs services Amazon ECS ne peuvent pas être créés en parallèle ou simultanément. Vous devez ajouter la valeur `dependsOn` au service client frontal pour chaque service backend client-serveur auquel les tâches clientes se connectent.

Ensuite, supposons qu'un service frontal existe sans configuration Service Connect. Les tâches se connectent à un service backend existant. Ajoutez d'abord une configuration Service Connect client-serveur au service backend, en utilisant le même nom dans le DNS ou `clientAlias` que celui utilisé par le frontend. Cela crée un nouveau déploiement, de sorte que tous les déploiements annulent la détection ou Console de gestion AWS, AWS CLI, AWS SDK et autres méthodes

permettant de revenir en arrière et de rétablir le service backend au déploiement et à la configuration précédents. Si vous êtes satisfait des performances et du comportement du service backend, ajoutez une configuration Service Connect client ou client-serveur au service frontal. Seules les tâches du nouveau déploiement utilisent le proxy Service Connect qui est ajouté à ces nouvelles tâches. Si vous rencontrez des problèmes avec cette configuration, vous pouvez revenir en arrière et revenir à votre configuration précédente en utilisant la détection d'annulation du déploiement ou Console de gestion AWS, AWS CLI, AWS SDK et autres méthodes permettant de revenir en arrière et de rétablir le service backend au déploiement et à la configuration précédents. Si vous utilisez un autre système de découverte de services basé sur le DNS au lieu de Service Connect, toutes les applications frontales ou clientes commencent à utiliser de nouveaux points de terminaison et à modifier la configuration des points de terminaison après l'expiration du cache DNS local, ce qui prend généralement plusieurs heures.

Mise en réseau

Par défaut, le proxy Service Connect écoute sur le `containerPort` à partir du mappage de port de la définition de tâche. Les règles de votre groupe de sécurité doivent autoriser le trafic entrant (ingress) vers ce port depuis les sous-réseaux où les clients seront exécutés.

Même si vous définissez un numéro de port dans la configuration du service Service Connect, cela ne modifie pas le port du service client-serveur écouté par le proxy Service Connect. Lorsque vous définissez ce numéro de port, Amazon ECS modifie le port du point de terminaison auquel les services clients se connectent, sur le proxy Service Connect au sein de ces tâches. Le proxy du service client se connecte au proxy du service client-serveur à l'aide du `containerPort`.

Si vous souhaitez modifier le port sur lequel le proxy Service Connect écoute, modifiez le `ingressPortOverride` dans la configuration Service Connect du service client-serveur. Si vous modifiez ce numéro de port, vous devez autoriser le trafic entrant sur ce port utilisé par le trafic vers ce service.

Le trafic que vos applications envoient aux services Amazon ECS configurés pour Service Connect nécessite qu'Amazon VPC et les sous-réseaux disposent de règles de table de routage et de règles d'ACL réseau qui autorisent les numéros de port `containerPort` et `ingressPortOverride` que vous utilisez.

Vous pouvez utiliser Service Connect pour envoyer du trafic entre les VPC. Les mêmes exigences concernant les règles de table de routage, les ACL réseau et les groupes de sécurité s’appliquent aux deux VPC.

Par exemple, deux clusters créent des tâches dans différents VPC. Un service dans chaque cluster est configuré pour utiliser le même espace de noms. Les applications dans ces deux services peuvent résoudre tous les points de terminaison de l'espace de noms sans aucune configuration DNS VPC. Toutefois, les proxys ne peuvent se connecter que si l'appairage de VPC, les tables de routage du VPC ou du sous-réseau et les ACL du réseau VPC autorisent le trafic sur les numéros de port `containerPort` et `ingressPort0verride`.

Pour les tâches utilisant le mode réseau `bridge`, vous devez créer un groupe de sécurité doté d'une règle entrante autorisant le trafic sur la plage de ports dynamiques supérieure. Attribuez ensuite le groupe de sécurité à toutes les instances EC2 du cluster Service Connect.

Proxy Service Connect

Si vous créez ou mettez à jour un service avec la configuration Service Connect, Amazon ECS ajoute un nouveau conteneur à chaque nouvelle tâche au moment où elle est lancée. Ce modèle d'utilisation d'un conteneur séparé est nommé `sidecar`. Ce conteneur n'est pas présent dans la définition de la tâche et vous ne pouvez pas le configurer. Amazon ECS gère la configuration des conteneurs dans le service. Cela vous permet de réutiliser les mêmes définitions de tâches entre plusieurs services, espaces de noms et tâches sans Service Connect.

Ressource de proxy

- Pour les définitions de tâches, vous devez définir les paramètres d'UC et de mémoire.

  Nous vous recommandons d'ajouter 256 unités d'UC supplémentaires et au moins 64 Mio de mémoire à votre UC et à votre mémoire de tâche pour le conteneur proxy Service Connect. Dans AWS Fargate, la quantité de mémoire la plus faible que vous pouvez définir est de 512 Mio de mémoire. Sur Amazon EC2, la mémoire de définition des tâches est requise.
- Pour le service, vous définissez la configuration du journal dans la configuration Service Connect.
- Si vous prévoyez que les tâches de ce service reçoivent plus de 500 demandes par seconde à leur charge maximale, nous vous recommandons d'ajouter 512 unités de processeur à votre processeur de tâches dans cette définition de tâche pour le conteneur de proxy Service Connect.
- Si vous prévoyez de créer plus de 100 services Service Connect dans l'espace de noms ou 2 000 tâches au total pour tous les services Amazon ECS dans l'espace de noms, nous vous recommandons d'ajouter 128 Mio de mémoire à votre mémoire de tâche pour le conteneur de proxy Service Connect. Vous devez le faire dans chaque définition de tâche utilisée par tous les services Amazon ECS dans l'espace de noms.

Configuration du proxy

Vos applications se connectent au proxy dans le conteneur sidecar dans le cadre de la même tâche dans laquelle se trouve l'application. Amazon ECS configure la tâche et les conteneurs de telle sorte que les applications ne se connectent au proxy que lorsque l’application est connectée aux noms des points de terminaison dans le même espace de noms. Tout le reste du trafic n'utilise pas le proxy. L'autre trafic inclut des adresses IP dans le même VPC, AWS points de terminaison de service et trafic externe.

Équilibrage de charge

Service Connect configure le proxy pour qu'il utilise la stratégie alternée pour équilibrer la charge entre les tâches d'un point de terminaison Service Connect. Le proxy local présent dans la tâche d'où provient la connexion choisit l'une des tâches du service client-serveur qui fournit le point de terminaison.

Par exemple, considérez une tâche qui s'exécute WordPress dans un service configuré en tant que service à la clientèle dans un espace de noms appelé local. Il existe un autre service avec deux tâches qui exécutent la base de données MySQL. Ce service est configuré pour fournir un point de terminaison appelé `mysql` via Service Connect dans le même espace de noms. Dans le WordPress tâche, le WordPress l'application se connecte à la base de données à l'aide du nom du point de terminaison. Les connexions à ce nom sont redirigées vers le proxy qui s’exécute dans un conteneur sidecar dans la même tâche. Le proxy peut ensuite se connecter à l'une ou l'autre des tâches MySQL en utilisant la stratégie alternée.

Stratégies d'équilibrage de charge : alternance

Détection des valeurs aberrantes

Cette fonctionnalité utilise les données dont dispose le proxy concernant les échecs de connexion précédents pour éviter d'envoyer de nouvelles connexions aux hôtes dont les connexions ont échoué. Service Connect configure la fonctionnalité de détection des valeurs aberrantes du proxy afin de fournir des surveillances de l'état passives.

En reprenant l’exemple précédent, le proxy peut se connecter à l’une ou l’autre des tâches MySQL. Si le proxy a établi plusieurs connexions à une tâche MySQL spécifique et que 5 connexions ou plus ont échoué au cours des 30 dernières secondes, le proxy évite cette tâche MySQL pendant 30 à 300 secondes.

Nouvelles tentatives

Service Connect configure le proxy pour qu'il tente à nouveau la connexion qui passe par le proxy et échoue, et la deuxième tentative évite d'utiliser l'hôte de la connexion précédente. Cela garantit que chaque connexion via Service Connect n'échoue pas pour des raisons ponctuelles.

Nombre de nouvelles tentatives : 2

Timeout

Service Connect configure le proxy pour qu'il attende au maximum que vos applications client-serveur répondent. La valeur de délai d'attente par défaut est de 15 secondes, mais elle peut être mise à jour.

Paramètres facultatifs :

idleTimeout : durée en secondes pendant laquelle une connexion reste active lorsqu'elle est au repos. Une valeur de `0` désactive `idleTimeout`.

La valeur par défaut de `idleTimeout` pour `HTTP/HTTP2/GRPC` est de cinq minutes.

La valeur par défaut de `idleTimeout` pour `TCP` est d'une heure.

parRequestTimeout - Le temps d'attente pour que l'amont réponde avec une réponse complète par demande. Une valeur de `0` désactive `perRequestTimeout`. Ce paramètre ne peut être défini que lorsque le protocole `appProtocol` du conteneur d'application est `HTTP/HTTP2/GRPC`. La valeur par défaut est de 15 secondes.

> **Note**
>
> Si `idleTimeout` est défini sur une durée inférieure à `perRequestTimeout`, la connexion se ferme lorsque le `idleTimeout` est atteint et non le `perRequestTimeout`.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez Service Connect :

- Les tâches exécutées dans Fargate doivent utiliser la version de plateforme Linux Fargate `1.4.0` ou supérieure pour pouvoir utiliser Service Connect.

- La version de l'agent Amazon ECS sur l'instance de conteneur doit être la version `1.67.2` ou supérieure.
- Les instances de conteneurs doivent exécuter Amazon ECS-optimized Version AMI Amazon Linux 2023 `20230428` ou version ultérieure, ou Amazon ECS-optimized Version AMI d'Amazon Linux 2 `2.0.20221115` pour utiliser Service Connect. Ces versions disposent de l'agent Service Connect en plus de l'agent de conteneur Amazon ECS. Pour plus d'informations sur l'agent Service Connect, consultez Agent Amazon ECS Service Connect sur GitHub.
- Les instances de conteneur doivent disposer de l'autorisation `ecs:Poll` pour la ressource `arn:aws:ecs:region:0123456789012:task-set/cluster/*`. Si vous utilisez le `ecsInstanceRole`, vous n'avez pas besoin d'ajouter d'autorisations supplémentaires. La stratégie gérée par `AmazonEC2ContainerServiceforEC2Role` dispose de ces autorisations. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
- Les tâches qui utilisent le mode réseau `bridge` et utilisent Service Connect ne prennent pas en charge le paramètre de définition du conteneur `hostname`.
- Les définitions de tâches doivent définir la limite de mémoire des tâches pour utiliser Service Connect. Pour plus d'informations, consultez Proxy Service Connect.
- Les définitions de tâches qui définissent les limites de mémoire des conteneurs ne sont pas prises en charge.

  Vous pouvez définir des limites de mémoire de conteneur sur vos conteneurs, mais vous devez définir la limite de mémoire des tâches sur un nombre supérieur à la somme des limites de mémoire des conteneurs. Le processeur et la mémoire supplémentaires inclus dans les limites de tâches que vous n'allouez pas dans les limites de conteneur sont utilisés par le conteneur de proxy Service Connect et les autres conteneurs qui ne définissent pas de limites de conteneur. Pour plus d'informations, consultez Proxy Service Connect.
- Vous pouvez configurer Service Connect pour utiliser n'importe quel AWS Cloud Map espace de noms dans la même région qui se trouvent tous dans la même Compte AWS ou sont partagés avec votre Compte AWS en utilisant AWS Resource Access Manager. Pour plus d'informations sur l'utilisation d'espaces de nom partagés, consultez la section Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms.
- Chaque service ne peut appartenir qu'à un seul espace de noms.
- Seules les tâches créées par les services sont prises en charge.
- Tous les points de terminaison doivent être uniques dans un espace de noms.
- Tous les noms de découverte doivent être uniques dans un espace de noms.

- Vous devez redéployer les services existants avant que les applications puissent résoudre les nouveaux points de terminaison. Les nouveaux points de terminaison ajoutés à l'espace de noms après le déploiement le plus récent ne seront pas ajoutés à la configuration de la tâche. Pour plus d'informations, consultez the section called "Composants de Service Connect".
- Service Connect ne supprime pas les espaces de noms lorsque les clusters sont supprimés. Vous devez supprimer les espaces de noms dans AWS Cloud Map.
- Le trafic de l'Application Load Balancer est acheminé par défaut via l'agent Service Connect en mode réseau `awsvpc`. Si vous souhaitez que le trafic non lié au service contourne l'agent Service Connect, utilisez le paramètre `ingressPortOverride` dans la configuration de votre service Service Connect.
- Service Connect avec TLS ne prend pas en charge le mode réseau `bridge`. Seul le mode réseau `awsvpc` est pris en charge.
- Dans `awsvpc` mode, le proxy Service Connect redirige le trafic vers l'hôte local IPv4 `127.0.0.1` pour les services en IPv4-only et configurations à double pile. Pour les services en IPv6-only configuration, le proxy transfère le trafic vers l'hôte local IPv6 `::1`. Nous vous recommandons de configurer vos applications pour qu'elles écoutent les deux hôtes locaux pour une expérience fluide lors de la mise à jour d'un service depuis IPv4-only ou une configuration dualstack pour IPv6-only. Pour plus d'informations, consultez Options de mise en réseau des tâches Amazon ECS pour EC2 et Options de mise en réseau des tâches Amazon ECS pour Fargate.

Service Connect ne prend pas en charge les fonctionnalités suivantes :

- Conteneurs Windows
- HTTP 1.0
- Tâches autonomes
- Les services qui utilisent blue/green alimenté par CodeDeploy et types de déploiements externes
- L'instance de conteneur `External` pour Amazon ECS Anywhere n'est pas prise en charge par Service Connect.
- PPv2
- Mode FIPS

Vue d'ensemble de la configuration d'Amazon ECS Service Connect

Lorsque vous utilisez Service Connect, vous devez configurer certains paramètres dans vos ressources.

Le tableau suivant décrit les paramètres de configuration pour les ressources Amazon ECS.

Emplacement des paramètres	Type d'application	Description	Obligatoire
Définition de tâche	Client	Aucune modification n'est disponible pour Service Connect dans les définitions des tâches du client.	N/A
Définition de tâche	Client-server	Les serveurs doivent ajouter des champs de `name` aux ports dans les `portMappings` des conteneurs. Pour plus d'informations, consultez `portMappings`	Oui
Définition de tâche	Client-server	Les serveurs peuvent éventuellement fournir un protocole d'application (par exemple, HTTP) pour recevoir des métriques spécifiques au protocole pour leurs applications serveur (par exemple, `HTTP 5xx`).	Non
Définition des services	Client	Les services clients doivent ajouter une `serviceConnectConfiguration` pour configurer l'espace de noms à rejoindre. Cet espace de noms doit contenir tous les services de serveur que ce service doit découvrir. Pour plus d'informations, consultez `serviceConnectConfiguration`.	Oui
Définition des services	Client-server	Les services de serveur doivent ajouter une `serviceConnectConfiguration` pour configurer les noms DNS, les numéros de port et l'espace de noms à partir desquels le service est disponible. Pour plus d'informations,	Oui

Emplacement des paramètres	Type d'application	Description	Obligatoire
		consultez `serviceConnectConfiguration`.	
Cluster	Client	Les clusters peuvent ajouter un espace de noms Service Connect par défaut. Les nouveaux services du cluster héritent de l'espace de noms lorsque Service Connect est configuré dans un service.	Non
Cluster	Client-server	Aucune modification n'est disponible pour Service Connect dans les clusters qui s'appliquent aux services de serveur. Les définitions de tâches de serveur et les services doivent définir la configuration correspondante.	N/A

Vue d'ensemble des étapes de configuration de Service Connect

Les étapes suivantes présentent la configuration de Service Connect.

> **Important**
>
> - Service Connect crée AWS Cloud Map services sur votre compte. Les modifier AWS Cloud Map ressources manuellement registering/deregistering instances, la modification des attributs d'instance ou la suppression d'un service peuvent entraîner un comportement inattendu pour le trafic de vos applications ou les déploiements ultérieurs.
> - Service Connect ne prend pas en charge les liens dans la définition des tâches.

1. Ajoutez des noms de port aux mappages de port dans vos définitions de tâches. Vous pouvez également identifier le protocole de couche 7 de l'application afin d'obtenir des métriques supplémentaires.

2. Créez un cluster avec AWS Cloud Map espace de noms, utilisez un espace de noms partagé ou créez l'espace de noms séparément. Pour simplifier l'organisation, créez un cluster avec le nom que vous souhaitez pour l'espace de noms et spécifiez le même nom pour l'espace de noms. Dans ce cas, Amazon ECS crée un nouvel espace de noms HTTP avec la configuration nécessaire. Service Connect n'utilise ni ne crée de zones hébergées DNS dans Amazon Route 53.
3. Configurez les services pour créer des points de terminaison Service Connect dans l'espace de noms.
4. Déployez des services pour créer les points de terminaison. Amazon ECS ajoute un conteneur de proxy Service Connect à chaque tâche et crée les points de terminaison Service Connect dans AWS Cloud Map. Ce conteneur n'est pas configuré dans la définition de tâche, et cette dernière peut être réutilisée sans modification pour créer plusieurs services dans le même espace de noms ou dans plusieurs espaces de noms.
5. Déployez des applications clientes en tant que services pour vous connecter aux points de terminaison. Amazon ECS les connecte aux points de terminaison Service Connect par le proxy Service Connect dans chaque tâche.

   Les applications utilisent le proxy uniquement pour se connecter aux points de terminaison Service Connect. Il n'existe aucune configuration supplémentaire pour utiliser le proxy. Le proxy effectue un équilibrage de charge alterné, une détection des valeurs aberrantes et de nouvelles tentatives. Pour plus d'informations sur le proxy, veuillez consulter Proxy Service Connect.
6. Surveillez le trafic via le proxy Service Connect d'Amazon CloudWatch.

Configuration du cluster

Vous pouvez définir un espace de noms par défaut pour Service Connect lorsque vous créez ou mettez à jour le cluster. Le nom de l'espace de noms que vous spécifiez par défaut peut être identique Région AWS et compte, ou dans le même Région AWS et partagé par un autre Compte AWS en utilisant AWS Resource Access Manager.

Si vous créez un cluster et que vous spécifiez un espace de noms Service Connect par défaut, le cluster attend dans l'état `PROVISIONING` pendant qu'Amazon ECS crée l'espace de noms. Vous pouvez voir un `attachment` dans l'état du cluster qui indique l'état de l'espace de noms. Les pièces jointes ne sont pas affichées par défaut dans AWS CLI, vous devez ajouter `--include ATTACHMENTS` pour les voir.

Si vous souhaitez utiliser un espace de noms partagé avec votre Compte AWS en utilisant AWS RAM, spécifiez l'Amazon Resource Name (ARN) de l'espace de noms dans la configuration du

cluster. Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms.

Configuration du service

Service Connect est conçu pour nécessiter une configuration minimale. Vous devez définir un nom pour chaque mappage de port que vous souhaitez utiliser avec Service Connect dans la définition de tâche. Dans le service, vous devez activer Service Connect et sélectionner un espace de noms dans votre Compte AWS ou un espace de noms partagé pour créer un service client. Pour créer un service client-serveur, vous devez ajouter une configuration de service Service Connect unique qui correspond au nom de l'un des mappages de port. Amazon ECS réutilise le numéro de port et le nom de port figurant dans la définition de tâche pour définir le service et le point de terminaison Service Connect. Pour remplacer ces valeurs, vous pouvez utiliser les autres paramètres Discovery (Découverte), DNS et Port dans la console ou `discoveryName` et `clientAliases`, respectivement, dans l'API Amazon ECS.

Configuration initiale du bilan de santé

Service Connect s'assure que les tâches sont saines avant d'acheminer le trafic vers celles-ci. Lorsqu'une tâche est lancée (pendant les déploiements, la mise à l'échelle ou les remplacements), Service Connect surveille l'état de votre tâche pour s'assurer qu'elle est prête à accepter le trafic. Vous devez définir des contrôles de santé pour le conteneur essentiel dans votre définition de tâche pour activer ce comportement.

Le comportement du bilan de santé initial tient compte des retards potentiels avant d'atteindre l'état où une tâche est prête à accepter du trafic :

- Si une tâche est `HEALTHY`, il est immédiatement disponible pour le trafic.
- Si l'état de santé d'une tâche est `UNKNOWN`, Service Connect suit la configuration du contrôle de santé du conteneur (voir the section called “Surveillance de l’état”) des conteneurs essentiels de la tâche pour calculer un délai d'attente, jusqu'à `8 minutes`, avant de le mettre à la disposition du trafic, même s'il reste `UNKNOWN`.
- Si une tâche est `UNHEALTHY`, Amazon ECS peut lancer des tâches de remplacement. Si aucune tâche saine n'est disponible, votre déploiement peut être annulé en fonction de la configuration de votre service.

Pour tout le trafic en cours, Service Connect utilise des bilans de santé passifs basés sur la détection des valeurs aberrantes afin d'acheminer le trafic de manière efficace.

Amazon ECS Service Connect avec service partagé AWS Cloud Map espaces de noms

Amazon ECS Service Connect prend en charge l'utilisation du partage AWS Cloud Map espaces de noms sur plusieurs Comptes AWS dans le même Région AWS. Cette fonctionnalité vous permet de créer des applications distribuées dans lesquelles les services s'exécutent dans différents Comptes AWS peuvent se découvrir et communiquer entre eux via Service Connect. Les espaces de noms partagés sont gérés à l'aide d' AWS Resource Access Manager (AWS RAM), qui permet un partage sécurisé des ressources entre comptes. Pour plus d'informations sur les espaces de noms partagés, consultez Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur.

> **Important**
>
> Vous devez utiliser l'autorisation gérée `AWSRAMPermissionCloudMapECSFullPermission` pour partager l'espace de noms afin que Service Connect fonctionne correctement avec l'espace de noms.

Lorsque vous utilisez le mode partagé AWS Cloud Map espaces de noms avec Service Connect, services provenant de plusieurs Comptes AWS peuvent participer au même espace de noms de service. Ceci est particulièrement utile pour les organisations ayant plusieurs Comptes AWS qui doivent maintenir une communication service à service au-delà des limites des comptes tout en préservant la sécurité et l'isolement.

> **Note**
>
> Pour communiquer avec des services situés dans différents VPC, vous devez configurer la connectivité entre VPC. Cela peut être réalisé à l'aide d'une connexion d'appairage de VPC. Pour plus d'informations, consultez la section Création ou suppression d'une connexion d'appairage de VPC dans le Guide d'appairage de VPC Amazon Virtual Private Cloud.

Utilisation du partage AWS Cloud Map espaces de noms avec Amazon ECS Service Connect

Configuration du partage AWS Cloud Map les espaces de noms pour Service Connect impliquent les étapes suivantes : le propriétaire de l'espace de noms crée l'espace de noms, le propriétaire le partage via AWS Resource Access Manager (AWS RAM), le consommateur acceptant le partage de ressources et le consommateur configurant Service Connect pour utiliser l'espace de noms partagé.

Étape 1 : Créez le AWS Cloud Map espace de nom

Le propriétaire de l'espace de noms crée un AWS Cloud Map espace de noms qui sera partagé avec d'autres comptes.

Pour créer un espace de noms à partager à l'aide du Console de gestion AWS

1. Ouvrez le AWS Cloud Map console à https://console.aws.amazon.com/cloudmap/.
2. Choisissez Create namespace (Créer un espace de noms).
3. Saisissez un Nom d’espace de noms. Ce nom sera utilisé par les services sur tous les comptes participants.
4. Pour Type d’espace de noms, choisissez le type adapté à votre cas d’utilisation :
   - Appels d’API : espaces de noms HTTP pour la découverte de service sans fonctionnalité DNS.
   - Appels d’API et requêtes DNS dans les VPC : espaces de noms DNS privés pour la découverte de service avec des requêtes DNS privées dans un VPC.
   - Appels d’API et requêtes DNS publiques : espaces de noms DNS publics pour la découverte de service avec des requêtes DNS publiques.
5. Choisissez Create namespace (Créer un espace de noms).

Étape 2 : partager l'espace de noms à l'aide de AWS RAM

Le propriétaire de l'espace de noms utilise AWS RAM pour partager l'espace de noms avec d'autres Comptes AWS.

Pour partager un espace de noms à l'aide du AWS RAM console

1. Ouvrez le AWS RAM console à https://console.aws.amazon.com/ram/.
2. Choisissez Créer une ressource.
3. Sous Nom, saisissez un nom descriptif pour le partage de ressources.
4. Dans la section Ressources :
   a. Pour Type de ressource, choisissez Espaces de noms Cloud Map.
   b. Sélectionnez l’espace de noms que vous avez créé à l’étape précédente.
5. Dans le Autorisations gérées section, spécifiez AWSRAMPermissionCloudMapECSFullPermission.

> **Important**
>
> Vous devez utiliser l'autorisation gérée `AWSRAMPermissionCloudMapECSFullPermission` pour partager l'espace de noms afin que Service Connect fonctionne correctement avec l'espace de noms.

6. Dans la section Principaux, spécifiez l' Comptes AWS avec lequel vous souhaitez partager l'espace de noms. Vous pouvez saisir des ID de compte ou des ID d'unité organisationnelle.
7. Choisissez Créer une ressource.

Étape 3 : accepter le partage de ressources

Les comptes consommateurs d'espaces de noms doivent accepter l'invitation de partage de ressources pour utiliser l'espace de noms partagé.

Pour accepter une invitation de partage de ressources à l'aide du AWS RAM console

1. Dans le compte client, ouvrez le AWS RAM console à https://console.aws.amazon.com/ram/.
2. Dans le volet de navigation, choisissez Partagé avec moi, puis Partages de ressources.
3. Sélectionnez l'invitation au partage de ressources, puis choisissez Accepter le partage de ressources.
4. Après avoir accepté, notez l'ARN de l'espace de noms partagé dans les détails de la ressource. Vous utiliserez cet ARN lors de la configuration des services Service Connect.

Étape 4 : configurer un service Amazon ECS avec un espace de noms partagé

Après avoir accepté l'espace de noms partagé, le consommateur de l'espace de noms peut configurer les services Amazon ECS pour utiliser l'espace de noms partagé. La configuration est similaire à l'utilisation d'un espace de noms normal, mais vous devez spécifier l'ARN de l'espace de noms au lieu du nom. Pour une procédure détaillée de création de service, consultez la section Création d'un déploiement de mise à jour propagée Amazon ECS.

Pour créer un service avec un espace de noms partagé à l'aide du Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service.

3. Sous Services, choisissez Créer.
4. Après avoir renseigné d'autres informations en fonction de votre charge de travail, dans la section Service Connect, sélectionnez Utiliser Service Connect.
5. Pour Espace de noms, saisissez l'ARN complet de l'espace de noms partagé.

   Le format ARN est le suivant : `arn:aws:servicediscovery:`*`region`*`:`*`account-id`*`:namespace/`*`namespace-id`*
6. Configurez les autres paramètres Service Connect selon les besoins de votre type de service (client ou client-serveur).
7. Terminez le processus de création de service.

Vous pouvez également configurer les services à l'aide du AWS CLI ou AWS SDK en spécifiant l'ARN de l'espace de noms partagé dans le `namespace` paramètre du `serviceConnectConfiguration`.

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-def \
    --service-connect-configuration '{
        "enabled": true,
        "namespace": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcdef1234567890",
        "services": [{
            "portName": "web",
            "discoveryName": "my-service",
            "clientAliases": [{
                "port": 80,
                "dnsName": "my-service"
            }]
        }]
    }'
```

Considérations

Tenez compte des points suivants lorsque vous utilisez le mode partagé AWS Cloud Map espaces de noms avec Service Connect :

- AWS RAM doit être disponible dans Région AWS où vous souhaitez utiliser l'espace de noms partagé.
- L'espace de noms partagé doit être dans le même Région AWS en tant que services et clusters Amazon ECS.
- Vous devez utiliser l'ARN de l'espace de noms, et non l'ID, lorsque vous configurez Service Connect avec un espace de noms partagé.
- Tous les types d'espaces de noms sont pris en charge : espaces de noms HTTP, DNS privé et DNS public.
- Si l'accès à un espace de noms partagé est révoqué, les opérations Amazon ECS qui nécessitent une interaction avec l'espace de noms (telles que `CreateService`, `UpdateService` et `ListServicesByNamespace`) échoueront. Pour plus d'informations sur la résolution des problèmes d'autorisation liés aux espaces de noms partagés, consultez la section Résolution des problèmes liés à Amazon ECS Service Connect avec un service partagé AWS Cloud Map espaces de noms.
- Pour la découverte de service à l'aide de requêtes DNS dans un espace de noms DNS privé partagé :
  - Le propriétaire de l'espace de noms devra appeler `create-vpc-association-authorization` avec l'ID de la zone hébergée privée associée à l'espace de noms et le VPC du consommateur.

    ```
    aws route53 create-vpc-association-authorization --hosted-zone-id Z1234567890ABC --
    vpc VPCRegion=us-east-1,VPCId=vpc-12345678
    ```

  - Le consommateur de l'espace de noms devra appeler `associate-vpc-with-hosted-zone` avec l'ID de la zone hébergée privée.

    ```
    aws route53 associate-vpc-with-hosted-zone --hosted-zone-id Z1234567890ABC --vpc
     VPCRegion=us-east-1,VPCId=vpc-12345678
    ```

- Seul le propriétaire de l'espace de noms peut gérer le partage des ressources.
- Les consommateurs de l'espace de noms peuvent créer et gérer des services au sein de l'espace de noms partagé, mais ne peuvent pas modifier l'espace de noms lui-même.
- Les noms de découverte doivent être uniques dans l'espace de noms partagé, quel que soit le compte qui crée le service.
- Les services de l'espace de noms partagé peuvent découvrir et se connecter à des services provenant d'autres AWS comptes ayant accès à l'espace de noms.

- Lorsque vous activez TLS pour Service Connect et utilisez un espace de noms partagé, l'autorité de certification (CA) CA privée AWS est limitée à l'espace de noms. Lorsque l'accès à l'espace de noms partagé est révoqué, l'accès à l'autorité de certification est interrompu.
- Lorsqu'ils travaillent avec un espace de noms partagé, les propriétaires d'espaces de noms et les consommateurs n'ont pas accès à Amazon multicompte CloudWatch métriques par défaut. Les mesures cibles sont publiées uniquement pour les comptes qui possèdent des services clients. Un compte propriétaire de services client n'a pas accès aux métriques reçues par un compte propriétaire de services client-serveur, et vice versa. Pour permettre l'accès aux statistiques entre comptes, configurez CloudWatch observabilité entre comptes. Pour plus d'informations sur la configuration de l'observabilité entre comptes, consultez CloudWatch observabilité entre comptes dans le Amazon CloudWatch Guide de l'utilisateur. Pour plus d'informations sur le CloudWatch statistiques pour Service Connect, voir Amazon ECS CloudWatch indicateurs.

Résolution des problèmes liés à Amazon ECS Service Connect avec un service partagé AWS Cloud Map espaces de noms

Utilisez les informations suivantes pour résoudre les problèmes liés au partage AWS Cloud Map espaces de noms et Service Connect. Pour plus d'informations sur la localisation des messages d'erreur, consultez la section Résolution des problèmes liés à Amazon ECS.

Des messages d'erreur liés à des problèmes d'autorisations apparaissent en raison d'autorisations manquantes ou si l'accès à l'espace de noms est révoqué.

> **Important**
>
> Vous devez utiliser l'autorisation gérée `AWSRAMPermissionCloudMapECSFullPermission` pour partager l'espace de noms afin que Service Connect fonctionne correctement avec l'espace de noms.

Le message d'erreur apparaît dans l'un des formats suivants :

Une erreur s'est produite (ClientException) lorsque vous appelez le <OperationName> <account-id><user-name>opération : L'utilisateur : arn:aws:iam : ::user/ n'est pas autorisé à exécuter : <ActionName> sur la ressource : <ResourceArn> car aucune politique basée sur les ressources n'autorise le <ActionName> action

Les scénarios suivants peuvent générer un message d'erreur dans ce format :

Échec de création ou de mise à jour du cluster

Ces problèmes se produisent lorsque des opérations Amazon ECS telles que `CreateCluster` ou `UpdateCluster` échec dû à un manque AWS Cloud Map autorisations. Les opérations nécessitent des autorisations pour les opérations suivantes : AWS Cloud Map actions :

- `servicediscovery:GetNamespace`

Assurez-vous que l'invitation à partager la ressource a été acceptée dans le compte consommateur et que l'ARN correct de l'espace de noms est utilisé dans la configuration Service Connect.

Échec de création ou de mise à jour du service

Ces problèmes se produisent lorsque des opérations Amazon ECS telles que `CreateService` ou `UpdateService` échec dû à un manque AWS Cloud Map autorisations. Les opérations nécessitent des autorisations pour les opérations suivantes : AWS Cloud Map actions :

- `servicediscovery:CreateService`
- `servicediscovery:GetNamespace`
- `servicediscovery:GetOperation` (pour créer un nouveau service AWS Cloud Map)
- `servicediscovery:GetService` (lorsqu'un service AWS Cloud Map existe déjà)

Assurez-vous que l'invitation à partager la ressource a été acceptée dans le compte consommateur et que l'ARN correct de l'espace de noms est utilisé dans la configuration Service Connect.

Échec de l'opération **`ListServicesByNamespace`**

Ce problème se produit lorsque l'opération Amazon ECS `ListServicesByNamespace` échoue. L'opération nécessite des autorisations pour les actions AWS Cloud Map suivantes :

- `servicediscovery:GetNamespace`

Pour résoudre ce problème :

- Vérifiez que le compte consommateur dispose de l'autorisation `servicediscovery:GetNamespace`.
- Utilisez l'ARN de l'espace de noms lorsque vous appelez l'API, et pas le nom.
- Assurez-vous que le partage de ressources est actif et que l'invitation a été acceptée.

L'utilisateur : <iam-user>n'est pas autorisé à effectuer : <ActionName> sur la ressource : <ResourceArn> avec un refus explicite dans une politique basée sur l'identité.

Les scénarios suivants peuvent générer un message d'erreur dans ce format :

La suppression du service échoue et reste bloquée dans l'état **`DRAINING`**

Ce problème se produit lorsque les opérations Amazon ECS `DeleteService` échouent en raison d'une autorisation `servicediscovery:DeleteService` manquante lorsque l'accès à l'espace de noms est révoqué. Le service peut sembler avoir été supprimé avec succès au départ, mais il restera bloqué dans l'état `DRAINING`. Le message d'erreur apparaît sous la forme d'un événement de service Amazon ECS.

Pour résoudre ce problème, le propriétaire de l'espace de noms doit partager l'espace de noms avec le compte consommateur afin de permettre la suppression du service.

Les tâches en service ne s'exécutent pas

Ce problème se produit lorsque les tâches ne démarrent pas en raison d'autorisations manquantes. Le message d'erreur apparaît sous la forme d'une erreur de tâche arrêtée. Pour plus d'informations, consultez Résolution des erreurs liées aux tâches Amazon ECS arrêtées.

Ce qui suit AWS Cloud Map des actions sont nécessaires pour exécuter une tâche :

- `servicediscovery:GetOperation`
- `servicediscovery:RegisterInstance`

Assurez-vous que le compte consommateur dispose des autorisations requises et que l'espace de noms partagé est accessible.

Les tâches ne s'arrêtent pas correctement ou restent bloquées dans l'état **`DEACTIVATING`** ou **`DEPROVISIONING`**

Ce problème se produit lorsque les tâches ne parviennent pas à se désinscrire du AWS Cloud Map service lors de l'arrêt en raison d'autorisations manquantes. L'erreur apparaît sous la forme d'un `statusReason` dans la pièce jointe de la tâche, qui peut être récupérée à l'aide de l'API `DescribeTasks`. Pour plus d'informations, voir DescribeTasks dans le Référence de l'API Amazon Elastic Container Service.

Ce qui suit AWS Cloud Map des actions sont nécessaires pour arrêter une tâche :

- `servicediscovery:DeregisterInstance`
- `servicediscovery:GetOperation`

Si l'accès à l'espace de noms partagé est révoqué, les tâches peuvent rester dans l'état `DEACTIVATING` ou `DEPROVISIONING` jusqu'à ce que l'accès à l'espace de noms soit rétabli. Demandez au propriétaire de l'espace de noms de rétablir l'accès à l'espace de noms.

Journaux d'accès à Amazon ECS Service Connect

Amazon ECS Service Connect prend en charge les journaux d'accès afin de fournir une télémétrie détaillée sur les demandes individuelles traitées par le proxy Service Connect. Les journaux d'accès complètent les journaux d'applications existants en capturant les métadonnées du trafic par demande, telles que les méthodes HTTP, les chemins, les codes de réponse, les indicateurs et les informations de synchronisation. Cela permet une meilleure observabilité des modèles de trafic au niveau des demandes et des interactions entre les services pour un dépannage et une surveillance efficaces.

Pour activer les journaux d'accès, spécifiez à la fois `logConfiguration` et `accessLogConfiguration` objets dans `serviceConnectConfiguration` objet. Vous pouvez configurer le format des journaux et déterminer si les journaux doivent inclure des paramètres de requête dans `accessLogConfiguration`. Les journaux sont transmis au groupe de journaux de destination par le pilote de journal spécifié dans le `logConfiguration`.

```
{
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "myapp.namespace",
        "services": [
            ...
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "my-envoy-log-group",
                "awslogs-region": "us-west-2",
                "awslogs-stream-prefix": "myapp-envoy-logs"
            }
        },
         "accessLogConfiguration": {
            "format": "TEXT",
            "includeQueryParameters": "ENABLED"
        }
    }
```

```
}
```

Considérations

Lorsque vous activez l'accès aux journaux d'accès, tenez compte des points suivants :

- Les journaux d'accès et les journaux des applications sont tous deux écrits dans `/dev/stdout`. Pour séparer les journaux d'accès des journaux des applications, nous vous recommandons d'utiliser le `awsfirelens` pilote de journal avec un pilote personnalisé Fluent Bit ou Fluentd configuration.
- Nous vous recommandons d'utiliser `awslogs` pilote de journal pour envoyer les journaux d'application et d'accès au même CloudWatch destination.
- les journaux d'accès sont pris en charge sur les services Fargate qui utilisent la version de la plateforme `1.4.0` et plus.
- Les paramètres de requête tels que les identifiants de demande et les jetons sont exclus des journaux d'accès par défaut. Pour inclure les paramètres de requête dans les journaux d'accès, définissez `includeQueryParameters` pour `"ENABLED"`.

Formats de journal d'accès

les journaux d'accès peuvent être formatés soit dans des dictionnaires au format JSON, soit dans des chaînes au format texte, avec des différences dans les opérateurs de commande pris en charge pour les différents types de journaux d'accès.

Journaux d'accès HTTP

Les opérateurs de commande suivants sont inclus par défaut pour les journaux HTTP :

Text

```
[%START_TIME%] "%REQ(:METHOD)% %REQ(X-ENVOY-ORIGINAL-PATH?:PATH)% %PROTOCOL%"
%RESPONSE_CODE% %BYTES_RECEIVED% %BYTES_SENT% %DURATION%
%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)% "%REQ(X-FORWARDED-FOR)%" "%REQ(USER-AGENT)%"
"%REQ(X-REQUEST-ID)%" "%REQ(:AUTHORITY)%" "%UPSTREAM_HOST%"\n
```

JSON

```
{
  "start_time": "%START_TIME%",
```

```
  "method": "%REQ(:METHOD)%",
  "path": "%REQ(X-ENVOY-ORIGINAL-PATH?:PATH)%",
  "protocol": "%PROTOCOL%",
  "response_code": "%RESPONSE_CODE%",
  "bytes_received": "%BYTES_RECEIVED%",
  "bytes_sent": "%BYTES_SENT%",
  "duration_ms": "%DURATION%",
  "upstream_service_time": "%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%",
  "forwarded_for": "%REQ(X-FORWARDED-FOR)%",
  "user_agent": "%REQ(USER-AGENT)%",
  "request_id": "%REQ(X-REQUEST-ID)%",
  "authority": "%REQ(:AUTHORITY)%",
  "upstream_host": "%UPSTREAM_HOST%"
}
```

Journaux d'accès HTTP2

Outre les opérateurs de commande inclus pour les journaux HTTP, les journaux HTTP2 incluent `%STREAM_ID%` opérateur par défaut.

Text

```
[%START_TIME%] "%REQ(:METHOD)% %REQ(X-ENVOY-ORIGINAL-PATH?:PATH)% %PROTOCOL%"
%RESPONSE_CODE% %BYTES_RECEIVED% %BYTES_SENT% %DURATION%
%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)% "%REQ(X-FORWARDED-FOR)%" "%REQ(USER-AGENT)%"
"%REQ(X-REQUEST-ID)%" "%REQ(:AUTHORITY)%" "%UPSTREAM_HOST%" "%STREAM_ID%"\n
```

JSON

```
{
  "start_time": "%START_TIME%",
  "method": "%REQ(:METHOD)%",
  "path": "%REQ(X-ENVOY-ORIGINAL-PATH?:PATH)%",
  "protocol": "%PROTOCOL%",
  "response_code": "%RESPONSE_CODE%",
  "bytes_received": "%BYTES_RECEIVED%",
  "bytes_sent": "%BYTES_SENT%",
  "duration": "%DURATION%",
  "upstream_service_time": "%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%",
  "forwarded_for": "%REQ(X-FORWARDED-FOR)%",
  "user_agent": "%REQ(USER-AGENT)%",
  "request_id": "%REQ(X-REQUEST-ID)%",
```

```
  "authority": "%REQ(:AUTHORITY)%",
  "upstream_host": "%UPSTREAM_HOST%",
  "stream_id": "%STREAM_ID%"
}
```

Journaux d'accès gRPC

Outre les opérateurs de commande inclus pour les journaux HTTP, les journaux d'accès gRPC incluent `%STREAM_ID%` et `%GRPC_STATUS()%`opérateur par défaut.

Text

```
[%START_TIME%] "%REQ(:METHOD)% %REQ(X-ENVOY-ORIGINAL-PATH?:PATH)% %PROTOCOL%"
%RESPONSE_CODE% %GRPC_STATUS()% %BYTES_RECEIVED% %BYTES_SENT% %DURATION%
%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)% "%REQ(X-FORWARDED-FOR)%" "%REQ(USER-AGENT)%"
"%REQ(X-REQUEST-ID)%" "%REQ(:AUTHORITY)%" "%UPSTREAM_HOST%" "%STREAM_ID%"\n
```

JSON

```
{
  "start_time": "%START_TIME%",
  "method": "%REQ(:METHOD)%",
  "path": "%REQ(X-ENVOY-ORIGINAL-PATH?:PATH)%",
  "protocol": "%PROTOCOL%",
  "response_code": "%RESPONSE_CODE%",
  "grpc_status": "%GRPC_STATUS()%",
  "bytes_received": "%BYTES_RECEIVED%",
  "bytes_sent": "%BYTES_SENT%",
  "duration": "%DURATION%",
  "upstream_service_time": "%RESP(X-ENVOY-UPSTREAM-SERVICE-TIME)%",
  "forwarded_for": "%REQ(X-FORWARDED-FOR)%",
  "user_agent": "%REQ(USER-AGENT)%",
  "request_id": "%REQ(X-REQUEST-ID)%",
  "authority": "%REQ(:AUTHORITY)%",
  "upstream_host": "%UPSTREAM_HOST%",
  "stream_id": "%STREAM_ID%"
}
```

Journaux d'accès TCP

Les opérateurs de commande suivants sont inclus par défaut dans les journaux d'accès TCP :

Text

```
[%START_TIME%] %DOWNSTREAM_REMOTE_ADDRESS% %DOWNSTREAM_REMOTE_PORT%
%BYTES_RECEIVED% %BYTES_SENT% %DURATION%
%CONNECTION_TERMINATION_DETAILS% %CONNECTION_ID%\n
```

JSON

```
{
  "start_time": "%START_TIME%",
  "downstream_remote_address": "%DOWNSTREAM_REMOTE_ADDRESS%",
  "downstream_remote_port": "%DOWNSTREAM_REMOTE_PORT%",s
  "bytes_received": "%BYTES_RECEIVED%",
  "bytes_sent": "%BYTES_SENT%",
  "duration": "%DURATION%",
  "connection_termination_details": "%CONNECTION_TERMINATION_DETAILS%",
  "connection_id": %CONNECTION_ID%
}
```

Pour plus d'informations sur ces opérateurs de commande, voir Opérateurs de commande dans la documentation d'Envoy.

Activation des journaux d'accès pour Amazon ECS Service Connect

Les journaux d'accès ne sont pas activés par défaut pour les services Amazon ECS qui utilisent Service Connect. Vous pouvez activer les journaux d'accès de la manière suivante.

Activez les journaux d'accès à l'aide du AWS CLI

La commande suivante montre comment activer les journaux d'accès pour un service Amazon ECS à l'aide de AWS CLI en spécifiant un `accessLogConfiguration` lorsque vous créez le service :

```
aws ecs create-service \
    --cluster my-cluster \
    --service-name my-service \
    --task-definition my-task-def \
    --service-connect-configuration '{
        "enabled": true,
        "namespace": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
abcdef1234567890",
        "services": [{
```

```
            "portName": "web",
            "discoveryName": "my-service",
            "clientAliases": [{
                "port": 80,
                "dnsName": "my-service"
            }]
        }],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "my-envoy-log-group",
                "awslogs-region": "us-west-2",
                "awslogs-stream-prefix": "myapp-envoy-logs"
            }
        },
         "accessLogConfiguration": {
            "format": "TEXT",
            "includeQueryParameters": "ENABLED"
        }
    }'
```

Activer les journaux d'accès à l'aide de la console

Pour une procédure détaillée de création de service, consultez la section Création d’un déploiement de mise à jour propagée Amazon ECS.

Pour créer un service avec un espace de noms partagé à l'aide du Console de gestion AWS

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster dans lequel créer le service.
3. Sous Services, choisissez Créer.
4. Après avoir renseigné d’autres informations en fonction de votre charge de travail, dans la section Service Connect, sélectionnez Utiliser Service Connect.
5. Configurez les paramètres de Service Connect selon les besoins de votre type de service (client ou client-serveur).
6. Élargir Configuration du journal d'accès. Pour Formater, choisissez soit JSON ou TEXT.
7. Pour inclure les paramètres de requête dans les journaux d'accès, sélectionnez Inclure les paramètres de requête.
8. Terminez le processus de création de service.

Routage tenant compte des zones Amazon ECS Service Connect

Grâce au routage tenant compte des zones d'Amazon ECS Service Connect, le trafic reste dans la même zone de disponibilité (AZ) que le client. Cela réduit les coûts de transfert de données inter-AZ et la latence sans nécessiter de modifications supplémentaires de l'infrastructure ou du code d'application. Lorsque les terminaux sont équilibrés entre les zones de disponibilité, plus de 80 % du trafic reste local, tandis que les pondérations du trafic s'ajustent dynamiquement à mesure que les terminaux évoluent afin de maintenir une charge équilibrée entre les services cibles. Zone-aware le routage est activé par défaut pour tous les services nouveaux et existants qui utilisent Service Connect.

Comment fonctionne le routage tenant compte des zones

L'algorithme utilise la fonction de routage sensible à la zone Envoy dans le proxy Service Connect pour acheminer le trafic en fonction de la localisation du point de terminaison :

1. Découverte des terminaux - Le proxy découvre tous les points de terminaison disponibles pour le service de destination et leur emplacement AZ.
2. Same-AZ priorisation - Le proxy calcule le pourcentage du trafic qui reste dans la même zone de disponibilité en fonction du ratio de distribution des points de terminaison entre les zones de zone de zone source (client) et de destination (serveur). Lorsque les terminaux sont répartis de manière uniforme entre les zones, le trafic sur la même zone AZ dépasse généralement 80 %.
3. Routage de la capacité résiduelle - Le trafic qui ne peut pas rester dans la même zone de disponibilité est distribué aux autres zones de zone de trafic en fonction de leur capacité résiduelle. Le proxy calcule la capacité résiduelle de chaque zone et répartit le trafic inter-AZ proportionnellement entre les zones présentant une capacité résiduelle positive. À mesure que les terminaux augmentent ou diminuent, le proxy recalcule les poids de routage en temps réel.
4. Protection contre les surcharges et solution de repli - Pour éviter de surcharger une seule zone de disponibilité, le proxy a besoin d'au moins 2 fois plus de points de terminaison disponibles dans le service de destination. Pour une région 3-AZ, cela signifie au moins 6 points de terminaison. En dessous de ce seuil, le routage tenant compte des zones est désactivé et le trafic est réparti de manière uniforme grâce à un équilibrage de charge circulaire. Le routage se réactive automatiquement lorsque le nombre de points de terminaison dépasse le seuil. Si les terminaux d'une même zone de disponibilité deviennent indisponibles, le trafic est redistribué entre les autres zones de disponibilité afin de maintenir la disponibilité.

Vérifiez le routage tenant compte des zones sur Amazon EC2

Sur les instances de conteneur Amazon EC2 qui utilisent le moteur d'exécution Docker, vous pouvez consulter les statistiques du proxy Envoy dans le conteneur de l'agent Service Connect pour confirmer que le routage tenant compte des zones est actif. Vos instances de conteneur doivent utiliser un ECS-optimized Version AMI `20260323` ou plus tard. Cette procédure nécessite AWS Systems Manager Accès à l'hôte du gestionnaire de session et `docker exec`. Pour AWS Fargate ou des instances de conteneur qui utilisent `containerd`, utilisez Amazon Virtual Private Cloud Flow Logs pour vérifier le comportement de routage, comme décrit dans the section called “Surveillance du routage tenant compte des zones”.

Pour vérifier le routage tenant compte des zones sur Amazon EC2 avec Docker

1. Connectez-vous à l'instance de conteneur avec AWS Systems Manager Gestionnaire de sessions. Remplacer *instance-id* avec l'ID de votre instance de conteneur.

   ```
   aws ssm start-session --target instance-id
   ```

2. Ouvrez un shell dans le conteneur de l'agent Service Connect.

   ```
   sudo docker exec -it $(sudo docker ps --filter "name=ecs-service-connect" -q | head
    -1) /bin/sh
   ```

3. Interrogez les statistiques de routage des zones depuis l'interface d'administration d'Envoy.

   ```
   curl --unix-socket /tmp/envoy_admin.sock http://unix/stats | grep "lb_zone"
   ```

4. Vérifiez la sortie. Un déploiement en bonne santé avec un routage sensible aux zones actif n'entraîne aucune demande entre zones :

   ```
   cluster.my-service.lb_zone_routing_cross_zone: 0
   cluster.my-service.lb_zone_cluster_too_small: 0
   ```

Le tableau suivant décrit les mesures de routage des zones clés.

Métrique	Description
`lb_zone_routing_cross_zone`	Nombre de demandes acheminées vers un point de terminaison situé dans une autre

Métrique	Description
	zone de disponibilité. Une valeur continue de 0 confirme que tout le trafic reste dans la même zone de zone de trafic.
`lb_zone_cluster_too_small`	Nombre de fois où le routage tenant compte des zones a été contourné parce que le nombre de points de terminaison était inférieur au seuil minimum. Non-zero les valeurs lors du déploiement initial sont attendues et disparaissent lorsque les terminaux redeviennent sains. Un déploiement stable affiche 0.

Surveillance du routage tenant compte des zones

Utilisez les journaux de flux Amazon Virtual Private Cloud avec `az-id` champ permettant d'observer les modèles de trafic au niveau du réseau. Cela indique de quelle zone de trafic provient et se termine chaque enregistrement de flux, ce qui vous permet de mesurer les ratios de trafic de même zone de zone de couverture par rapport aux ratios de trafic inter-AZ. Vous pouvez également utiliser AWS Cost Explorer permet de suivre les frais de transfert de données inter-AZ avant et après le redéploiement de vos services afin de permettre un routage tenant compte des zones.

Pour plus d'informations sur la configuration des journaux de flux, voir Journaux de flux VPC dans le Guide de l'utilisateur d'Amazon Virtual Private Cloud.

Considérations

Lorsque vous utilisez le routage tenant compte des zones, tenez compte des points suivants :

- Zone-aware le routage fonctionne avec tous les types de lancements.
- Zone-aware le routage est compatible avec les espaces de noms Service Connect inter-comptes partagés via AWS Resource Access Manager.
- Les services existants (client et serveur) nécessitent un redéploiement unique pour activer le routage tenant compte des zones. Après le redéploiement initial, le routage s'ajuste dynamiquement à mesure que les points de terminaison changent sans autres redéploiements.
- Pour le type de lancement Amazon EC2, vos instances de conteneur doivent utiliser un Amazon ECS-optimized Version AMI `20260323` ou plus tard.

Chiffrement du trafic Amazon ECS Service Connect

Amazon ECS Service Connect prend en charge le chiffrement automatique du trafic avec des certificats du protocole TLS (Transport Layer Security) pour les services Amazon ECS. Lorsque vous pointez vos services Amazon ECS vers un Autorité de certification privée AWS (CA privée AWS), Amazon ECS fournit automatiquement des certificats TLS pour chiffrer le trafic entre vos services Amazon ECS Service Connect. Amazon ECS génère, fait pivoter et distribue les certificats TLS utilisés pour le chiffrement du trafic.

Le chiffrement automatique du trafic avec Service Connect utilise des fonctionnalités de chiffrement de pointe pour sécuriser vos communications entre services afin de répondre à vos exigences en matière de sécurité. Il prend en charge Autorité de certification privée AWS Certificats TLS avec `256-bit ECDSA` et `2048-bit RSA` chiffrement. Vous avez également un contrôle total sur les certificats privés et les clés de signature pour vous aider à respecter les exigences de conformité. Par défaut, TLS 1.3 est pris en charge, mais pas TLS 1.0 à 1.2. Service Connect prend en charge TLS 1.3 avec les chiffrements suivants :

- `TLS_AES_128_GCM_SHA256`
- `TLS_AES_256_GCM_SHA384`
- `TLS_CHACHA20_POLY1305_SHA256`

> **Note**
>
> Pour utiliser TLS 1.3, vous devez l'activer sur l'écouteur de la cible.
> Seul le trafic entrant et sortant passant par l'agent Amazon ECS est chiffré.

Service Connect et surveillances de l'état de l'Application Load Balancer

Vous pouvez utiliser Service Connect avec les surveillances de l'état de l'Application Load Balancer et le chiffrement TLS 1.3.

Configuration de l'Application Load Balancer

Configurez l'Application Load Balancer avec les paramètres suivants :

- Configurer un écouteur TLS avec une politique de sécurité TLS 1.3 (telle que `ELBSecurityPolicy-TLS13-1-2-2021-06`). Pour plus d'informations, consultez la section Politiques de sécurité pour votre Application Load Balancer.

- Créez un groupe cible avec les paramètres suivants :
  - Définissez le protocole sur HTTPS.
  - Attachez le groupe cible à l'écouteur TLS.
  - Configurez le port de surveillance de l'état pour qu'il corresponde au port de conteneur de votre service Service Connect.

Configuration de Service Connect

Configurez un service avec les paramètres suivants :

- Configurez le service pour utiliser le mode réseau `awsvpc`, car le mode réseau `bridge` n'est pas pris en charge.
- Activez Service Connect pour le service.
- Configurez l'équilibreur de charge avec les paramètres suivants :
  - Spécifiez le groupe cible que vous avez configuré pour votre Application Load Balancer.
  - Définissez le port de conteneur pour qu'il corresponde au port de conteneur du service Service Connect TLS.
- Évitez de configurer `ingressPortOverride` pour le service. Pour plus d'informations, voir ServiceConnectService dans le Référence de l'API Amazon Elastic Container Service.

Considérations

Tenez compte des points suivants lorsque vous utilisez Application Load Balancer, le protocole TLS et Service Connect :

- Utilisez le mode réseau `awsvpc` plutôt que le mode réseau `bridge` pour les surveillances de l'état HTTPS lorsque vous utilisez Service Connect avec le chiffrement TLS. Les surveillances de l'état HTTP continueront de fonctionner avec le mode `bridge`.
- Configurez le port de surveillance de l'état du groupe cible pour qu'il corresponde au port de conteneur du service Service Connect, et non au port HTTPS par défaut (443).

Autorité de certification privée AWS certificats et Service Connect

Vous devez disposer du rôle IAM d'infrastructure. Pour plus d'informations sur ce rôle, consultez la section Rôle IAM d'infrastructure Amazon ECS.

Autorité de certification privée AWS modes pour Service Connect

Autorité de certification privée AWS peut fonctionner selon deux modes : à usage général et de courte durée.

- Usage général : certificats configurables avec n'importe quelle date d'expiration.
- Short-lived - Certificats d'une validité maximale de sept jours.

Amazon ECS prend en charge les deux modes, mais nous vous recommandons d'utiliser des certificats de courte durée. Par défaut, les certificats sont renouvelés tous les cinq jours, et leur utilisation en mode éphémère permet de réaliser des économies importantes par rapport aux certificats à usage général.

Service Connect ne prend pas en charge la révocation des certificats et utilise plutôt des certificats de courte durée avec une rotation fréquente des certificats. Vous avez le droit de modifier la fréquence de rotation, de désactiver ou de supprimer les secrets à l'aide de la rotation gérée dans Secrets Manager, mais cela peut avoir les conséquences suivantes.

- Fréquence de rotation plus courte - Une fréquence de rotation plus courte entraîne des coûts plus élevés en raison de CA privée AWS, AWS KMS et Secrets Manager et Auto Scaling sont confrontés à une charge de travail accrue liée à la rotation.
- Fréquence de rotation plus longue : les communications de vos applications échouent si la fréquence de rotation dépasse sept jours.
- Suppression du secret : la suppression du secret entraîne l'échec de la rotation et a un impact sur les communications des applications clientes.

En cas d'échec de votre rotation de secret, un événement `RotationFailed` est publié dans AWS CloudTrail. Vous pouvez également configurer un CloudWatch Alarme pour `RotationFailed`.

> **Important**
>
> N'ajoutez pas de réplicas de régions aux secrets. Cela empêche Amazon ECS de supprimer le secret, car Amazon ECS n'est pas autorisé à supprimer des régions de la réplication. Si vous avez déjà ajouté la réplication, exécutez la commande suivante.

```
aws secretsmanager remove-regions-from-replication \
 --secret-id SecretId \
```

```
--remove-replica-regions region-name
```

Autorités de certification subordonnées

Vous pouvez apporter n'importe quel CA privée AWS, root ou subordonné, à Service Connect TLS pour émettre des certificats d'entité finale pour les services. L'émetteur spécifié est considéré comme le signataire et la source de confiance dans tous les cas. Vous pouvez délivrer des certificats client pour différentes parties de votre application à partir de différentes autorités de certification subordonnées. Lorsque vous utilisez le AWS CLI, fournissez l'Amazon Resource Name (ARN) de l'autorité de certification pour établir la chaîne de confiance.

On-premises Autorités de certification

Pour utiliser votre autorité de certification sur site, vous devez créer et configurer une autorité de certification subordonnée dans Autorité de certification privée AWS. Cela garantit que tous les certificats TLS émis pour vos charges de travail Amazon ECS partagent la chaîne de confiance avec les charges de travail que vous exécutez sur site et sont en mesure de se connecter en toute sécurité.

> **Important**
>
> Ajoutez le requis étiquette `AmazonECSManaged : true` dans votre CA privée AWS.

Infrastructure en tant que code

Lorsque vous utilisez Service Connect TLS avec des outils d'infrastructure en tant que code (IaC), il est important de configurer correctement vos dépendances afin d'éviter des problèmes tels que le blocage des services. Votre AWS KMS le rôle IAM clé, s'il est fourni, et CA privée AWS les dépendances doivent être supprimées après votre service Amazon ECS.

Si l'espace de noms utilisé pour Service Connect est un espace de noms partagé, vous pouvez choisir d'utiliser un espace de noms partagé CA privée AWS ressource. Pour plus d'informations, consultez la section Joindre une politique d'accès intercompte dans le Guide de l'utilisateur Autorité de certification privée AWS .

Service Connect et Secrets Manager

Lorsque vous utilisez Amazon ECS Service Connect avec le chiffrement TLS, le service interagit avec Secrets Manager de la manière suivante :

Service Connect utilise le rôle d'infrastructure fourni pour créer des secrets dans Secrets Manager. Ces secrets sont utilisés pour stocker les clés privées associées à vos certificats TLS afin de chiffrer le trafic entre vos services Service Connect.

> **Warning**
>
> La création et la gestion automatiques de ces secrets par Service Connect rationalisent le processus de mise en œuvre du chiffrement TLS pour vos services. Toutefois, il est important de connaître les implications potentielles en matière de sécurité. Les autres rôles IAM disposant d'un accès en lecture à Secrets Manager peuvent accéder à ces secrets créés automatiquement. Cela pourrait exposer du matériel cryptographique sensible à des parties non autorisées, si les contrôles d'accès ne sont pas correctement configurés.
> Pour atténuer ce risque, appliquez les pratiques exemplaires suivantes :
>
> - Gérez et limitez soigneusement l'accès à Secrets Manager, en particulier pour les secrets créés par Service Connect.
> - Auditez régulièrement les rôles IAM et leurs autorisations afin de garantir le respect du principe du moindre privilège.

Lorsque vous accordez un accès en lecture à Secrets Manager, pensez à exclure les clés privées TLS créées par Service Connect. Vous pouvez le faire en utilisant une condition dans vos politiques IAM pour exclure les secrets dont les ARN correspondent au modèle :

```
"arn:aws:secretsmanager:::secret:ecs-sc!"
```

Exemple de politique IAM qui refuse l'action `GetSecretValue` à tous les secrets ayant le préfixe `ecs-sc!` :

JSON

```
{
    "Version":"2012-10-17",
```

```
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "secretsmanager:GetSecretValue",
            "Resource": "arn:aws:secretsmanager:*:*:secret:ecs-sc!*"
        }
    ]
}
```

> **Note**
>
> Il s'agit d'un exemple général qui devra peut-être être ajusté en fonction de votre cas d'utilisation spécifique et AWS configuration du compte. Testez toujours vos politiques IAM de manière approfondie pour vous assurer qu'elles fournissent l'accès prévu tout en préservant la sécurité.

La compréhension du fonctionnement de Service Connect avec Secrets Manager vous permet de renforcer la sécurité de vos services Amazon ECS tout en tirant parti des avantages du chiffrement TLS automatique.

Service Connect et AWS Key Management Service

Vous pouvez utiliser AWS Key Management Service pour chiffrer et déchiffrer vos ressources Service Connect. AWS KMS est un service géré par AWS où vous pouvez créer et gérer des clés cryptographiques qui protègent vos données.

Lors de l'utilisation AWS KMS avec Service Connect, vous pouvez choisir d'utiliser un AWS clé possédée qui AWS gère pour vous, ou vous pouvez choisir un existant AWS KMS clé. Vous pouvez également créer un nouveau AWS KMS clé à utiliser.

Utilisation de votre propre clé de chiffrement

Vous pouvez fournir vos propres documents clés ou utiliser un magasin de clés externe via AWS Key Management Service Importez votre propre clé dans AWS KMS, puis spécifiez l'Amazon Resource Name (ARN) de cette clé dans Amazon ECS Service Connect.

Voici un exemple AWS KMS politique. Remplacez le *`user input`* valeurs avec les vôtres.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "id",
      "Effect": "Allow",
      "Principal": {
        "AWS": "arn:aws:iam::111122223333:role/role-name"
      },
      "Action": [
        "kms:Encrypt",
        "kms:Decrypt",
        "kms:GenerateDataKey",
        "kms:GenerateDataKeyPair"
      ],
      "Resource": "*"
    }
  ]
}
```

Pour de plus amples informations sur les stratégies de clé, consultez la section Création de stratégies de clé dans le Guide du développeur AWS Key Management Service .

> **Note**
>
> Service Connect ne prend en charge que le chiffrement symétrique AWS KMS clés. Vous ne pouvez utiliser aucun autre type de AWS KMS clé pour chiffrer vos ressources Service Connect. Pour vous aider à déterminer si AWS KMS la clé est une clé de chiffrement symétrique, voir Identifier les clés KMS asymétriques.

Pour plus d'informations sur AWS Key Management Service clés de chiffrement symétriques, voir Chiffrement symétrique AWS KMS clés dans le AWS Key Management Service Guide du développeur.

Activation du protocole TLS pour Amazon ECS Service Connect

Vous activez le chiffrement du trafic lorsque vous créez ou mettez à jour un service Service Connect.

Pour activer le chiffrement du trafic pour un service dans un espace de noms existant à l'aide du Console de gestion AWS

1. Vous devez disposer du rôle IAM d'infrastructure. Pour plus d'informations sur ce rôle, consultez la section Rôle IAM d'infrastructure Amazon ECS.
2. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
3. Dans le panneau de navigation, choisissez Namespaces (Espaces de noms).
4. Choisissez l'espace de noms du service pour lequel vous souhaitez activer le chiffrement du trafic.
5. Choisissez le service pour lequel vous souhaitez activer le chiffrement du trafic.
6. Choisissez Service de mise à jour et faites défiler la page jusqu'à la section Service Connect.
7. Choisissez Activer le chiffrement du trafic sous les informations de service pour activer le protocole TLS.
8. Pour Rôle Service Connect TLS, choisissez un rôle IAM d'infrastructure existant ou créez-en un.
9. Pour Signataire de l'autorité de certification, choisissez une autorité de certification existante ou créez-en une nouvelle.

   Pour plus d'informations, consultez Autorité de certification privée AWS certificats et Service Connect.
10. Pour Choisissez un AWS KMS key, choisissez un AWS clé détenue et gérée ou vous pouvez choisir une autre clé. Vous pouvez également choisir d'en créer une.

Pour un exemple d'utilisation du AWS CLI pour configurer le protocole TLS pour votre service, Configuration d'Amazon ECS Service Connect à l'aide du AWS CLI.

Vérification de l'activation du protocole TLS pour Amazon ECS Service Connect

Service Connect initie le protocole TLS au niveau de l'agent Service Connect et le résilie au niveau de l'agent de destination. Par conséquent, le code de l'application ne détecte jamais les interactions TLS. Procédez comme suit pour vérifier que TLS est activé.

1. Incluez la CLI `openssl` dans l'image de votre application.
2. Activez ECS Exec sur vos services pour vous connecter à vos tâches via SSM. Vous pouvez également lancer une instance Amazon EC2 dans le même Amazon VPC que le service.

3. Récupérez l'adresse IP et le port d'une tâche auprès d'un service que vous souhaitez vérifier. Vous pouvez récupérer l'adresse IP de la tâche dans le AWS Cloud Map console. Les informations se trouvent sur la page des détails du service correspondant à l'espace de noms.
4. Connectez-vous à l'une de vos tâches en utilisant `execute-command` comme dans l'exemple suivant. Vous pouvez également vous connecter à l'instance Amazon EC2 créée à l'étape 2.

```
$ aws ecs execute-command --cluster cluster-name \
    --task task-id  \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

> **Note**
>
> L'appel direct du nom DNS ne révèle pas le certificat.

5. Dans le shell connecté, utilisez la CLI `openssl` pour vérifier et afficher le certificat associé à la tâche.

   Exemple :

```
openssl s_client -connect 10.0.147.43:6379 < /dev/null 2> /dev/null \
| openssl x509 -noout -text
```

   Exemple de réponse :

```
Certificate:
    Data:
        Version: 3 (0x2)
        Serial Number:
            <serial-number>
        Signature Algorithm: ecdsa-with-SHA256
        Issuer: <issuer>
        Validity
            Not Before: Jan 23 21:38:12 2024 GMT
            Not After : Jan 30 22:38:12 2024 GMT
        Subject: <subject>
        Subject Public Key Info:
            Public Key Algorithm: id-ecPublicKey
                Public-Key: (256 bit)
```

```
                pub:
                    <pub>
                ASN1 OID: prime256v1
                NIST CURVE: P-256
        X509v3 extensions:
            X509v3 Subject Alternative Name:
                DNS:redis.yelb-cftc
            X509v3 Basic Constraints:
                CA:FALSE
            X509v3 Authority Key Identifier:
                keyid:<key-id>

            X509v3 Subject Key Identifier:
                1D:<id>
            X509v3 Key Usage: critical
                Digital Signature, Key Encipherment
            X509v3 Extended Key Usage:
                TLS Web Server Authentication, TLS Web Client Authentication
    Signature Algorithm: ecdsa-with-SHA256
        <hash>
```

Configuration du conteneur proxy Service Connect

Lorsqu'un service utilise Service Connect, Amazon ECS ajoute automatiquement un conteneur proxy à chaque tâche du service. Vous pouvez configurer les paramètres de ce conteneur proxy à l'aide du `containerConfigurations` liste dans `serviceConnectConfiguration` objet. Vos modifications prennent effet une fois que vous avez déployé le service.

Ces paramètres contrôlent le comportement du conteneur proxy lui-même, comme la verbosité des journaux que le proxy écrit sur ses propres opérations. Pour obtenir des métadonnées détaillées sur les demandes individuelles gérées par le proxy, utilisez plutôt les journaux d'accès de Service Connect. Pour plus d’informations, consultez Journaux d'accès à Amazon ECS Service Connect.

Définissez le niveau de journalisation du proxy

Le paramètre du niveau de journalisation contrôle la quantité de détails opérationnels que le conteneur proxy enregistre dans ses journaux d'applications. Si vous ne spécifiez pas de niveau de journalisation, le niveau par défaut est `info`.

Réglez le niveau de journalisation sur `debug` pour collecter des journaux proxy plus détaillés lorsque vous résolvez des problèmes de connectivité et de routage du trafic entre les services.

La journalisation du débogage augmente le volume de journaux écrits par le conteneur proxy. Service Connect transmet les journaux du proxy au groupe de journaux que vous avez défini dans `logConfiguration` de `serviceConnectConfiguration`.

> **Important**
>
> Rétablissez votre niveau de journalisation à `info` une fois le dépannage terminé pour éviter des coûts excessifs de génération de journaux et de stockage.

L'exemple suivant définit le niveau de journalisation du proxy sur `debug`.

```
{
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "myapp.namespace",
        "services": [
            ...
        ],
        "containerConfigurations": [
            {
                "logLevel": "debug"
            }
        ]
    }
}
```

Configuration d'Amazon ECS Service Connect à l'aide du AWS CLI

Vous pouvez créer un service Amazon ECS pour une tâche Fargate qui utilise Service Connect avec l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS depuis le AWS CLI, des kits de développement logiciel et l'API Amazon ECS sur IPv4 et IPv6. Pour plus d'informations, consultez Utilisation des points de terminaison à double pile Amazon ECS.

Conditions préalables

Les prérequis Service Connect sont les suivants :

- Vérifiez que la dernière version du AWS CLI est installé et configuré. Pour plus d'informations, consultez la section Installation ou mise à jour de la version la plus récente de l' AWS CLI.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.
- Vous avez un VPC, un sous-réseau, une table de routage et un groupe de sécurité prêts à être utilisés. Pour plus d'informations, consultez the section called "Créer un Virtual Private Cloud".
- Vous avez un rôle d'exécution de tâches portant le nom `ecsTaskExecutionRole` et la politique gérée par `AmazonECSTaskExecutionRolePolicy` est associée au rôle. Ce rôle permet à Fargate d'écrire les journaux des applications NGINX et les journaux du proxy Service Connect sur Amazon. CloudWatch Journaux. Pour plus d'informations, consultez Création du rôle d'exécution de tâche.

Étape 1 : créer le cluster

Pour créer votre espace de noms et votre cluster Amazon ECS, effectuez les étapes suivantes.

Pour créer le cluster Amazon ECS et AWS Cloud Map espace de nom

1. Créez un cluster Amazon ECS nommé `tutorial` que vous allez utiliser. Le paramètre `--service-connect-defaults` définit l'espace de noms par défaut du cluster. Dans l'exemple de sortie, un AWS Cloud Map espace de noms du nom `service-connect` n'existe pas dans ce compte et Région AWS, l'espace de noms est donc créé par Amazon ECS. L'espace de noms est créé dans AWS Cloud Map dans le compte, et il est visible avec tous les autres espaces de noms. Utilisez donc un nom descriptif.

   ```
   aws ecs create-cluster --cluster-name tutorial --service-connect-defaults
    namespace=service-connect
   ```

   Sortie :

   ```
   {
       "cluster": {
           "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
           "clusterName": "tutorial",
   ```

```
        "serviceConnectDefaults": {
            "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
        },
        "status": "PROVISIONING",
        "registeredContainerInstancesCount": 0,
        "runningTasksCount": 0,
        "pendingTasksCount": 0,
        "activeServicesCount": 0,
        "statistics": [],
        "tags": [],
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "capacityProviders": [],
        "defaultCapacityProviderStrategy": [],
        "attachments": [
            {
                "id": "a1b2c3d4-5678-90ab-cdef-EXAMPLE11111",
                "type": "sc",
                "status": "ATTACHING",
                "details": []
            }
        ],
        "attachmentsStatus": "UPDATE_IN_PROGRESS"
    }
}
}
```

2. Vérifiez que le cluster est créé :

```
aws ecs describe-clusters --clusters tutorial
```

Sortie :

```
{
    "clusters": [
        {
            "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
            "clusterName": "tutorial",
```

```
            "serviceConnectDefaults": {
                "namespace": "arn:aws:servicediscovery:us-
west-2:123456789012:namespace/ns-EXAMPLE"
            },
            "status": "ACTIVE",
            "registeredContainerInstancesCount": 0,
            "runningTasksCount": 0,
            "pendingTasksCount": 0,
            "activeServicesCount": 0,
            "statistics": [],
            "tags": [],
            "settings": [],
            "capacityProviders": [],
            "defaultCapacityProviderStrategy": []
        }
    ],
    "failures": []
}
```

3. (Facultatif) Vérifiez que l'espace de noms est créé dans AWS Cloud Map. Vous pouvez utiliser le Console de gestion AWS ou le normal AWS CLI configuration telle qu'elle est créée dans AWS Cloud Map.

   Par exemple, utilisez l' AWS CLI :

   ```
   aws servicediscovery get-namespace --id ns-EXAMPLE
   ```

   Sortie :

   ```
   {
       "Namespace": {
           "Id": "ns-EXAMPLE",
           "Arn": "arn:aws:servicediscovery:us-west-2:123456789012:namespace/ns-
   EXAMPLE",
           "Name": "service-connect",
           "Type": "HTTP",
           "Properties": {
               "DnsProperties": {
                   "SOA": {}
               },
               "HttpProperties": {
   ```

```
                "HttpName": "service-connect"
            }
        },
        "CreateDate": 1661749852.422,
        "CreatorRequestId": "service-connect"
    }
}
```

Étape 2 : créer le service pour le serveur

La fonctionnalité Service Connect est destinée à interconnecter plusieurs applications sur Amazon ECS. Au moins l'une de ces applications doit fournir un service Web auquel se connecter. Au cours de cette étape, vous allez créer :

- La définition de tâche qui utilise l'image officielle non modifiée du conteneur NGINX et qui inclut la configuration de Service Connect.
- La définition du service Amazon ECS qui configure Service Connect pour fournir la découverte de service et la mise en proxy de maillage de services pour le trafic vers ce service. La configuration réutilise l'espace de noms par défaut de la configuration du cluster afin de réduire la quantité de configuration de service que vous effectuez pour chaque service.
- Le service Amazon ECS. Il exécute une tâche à l'aide de la définition de tâche et insère un conteneur supplémentaire pour le proxy Service Connect. Le proxy écoute sur le port depuis le mappage de ports du conteneur dans la définition de la tâche. Dans une application cliente exécutée dans Amazon ECS, le proxy intégré à la tâche client écoute les connexions sortantes vers le nom du port de définition de la tâche, le nom de découverte du service ou le nom d'alias du client de service, ainsi que le numéro de port provenant de l'alias client.

Pour créer le service Web avec Service Connect

1. Enregistrez une définition de tâche compatible avec Fargate et qui utilise le mode réseau `awsvpc`. Procédez comme suit :

    a. Créez un fichier nommé `service-connect-nginx.json` avec le contenu de la définition de tâche suivante.

    Cette définition de tâche configure Service Connect en ajoutant les paramètres de `name` et de `appProtocol` au mappage des ports. Le nom du port permet de mieux identifier ce

port dans la configuration du service lorsque plusieurs ports sont utilisés. Le nom du port est également utilisé par défaut comme nom détectable à utiliser par d'autres applications de l'espace de noms.

La définition de tâche contient le rôle IAM de tâche, car ECS Exec est activé sur le service.

> **Important**
>
> Cette définition de tâche utilise un `logConfiguration` pour envoyer la sortie nginx depuis `stdout` et `stderr` vers Amazon CloudWatch Journaux. Ce rôle d'exécution de tâches ne dispose pas des autorisations supplémentaires requises pour que CloudWatch Groupe de journaux. Créez le groupe de journaux dans CloudWatch Journaux à l'aide du Console de gestion AWS ou AWS CLI. Si vous ne souhaitez pas envoyer les journaux nginx à CloudWatch Journaux que vous pouvez supprimer `logConfiguration`.
> Remplacez le Compte AWS identifiant dans le rôle d'exécution des tâches avec votre Compte AWS identifiant.

```
{
    "family": "service-connect-nginx",
    "executionRoleArn": "arn:aws:iam::123456789012:role/ecsTaskExecutionRole",
    "taskRoleArn": "arn:aws:iam::123456789012:role/ecsTaskRole",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
        "name": "webserver",
        "image": "public.ecr.aws/docker/library/nginx:latest",
        "cpu": 100,
        "portMappings": [
            {
                "name": "nginx",
                "containerPort": 80,
                "protocol": "tcp",
                "appProtocol": "http"
            }
        ],
        "essential": true,
        "logConfiguration": {
```

```
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-nginx",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "nginx"
            }
        }
        }
    ],
    "cpu": "256",
    "memory": "512"
}
```

b. Enregistrez la définition de tâche à l'aide du fichier `service-connect-nginx.json` :

```
aws ecs register-task-definition --cli-input-json file://service-connect-nginx.json
```

2. Créez un service :

a. Créez un fichier nommé `service-connect-nginx-service.json` avec le contenu du service Amazon ECS que vous créez. Cet exemple utilise la définition de tâche créée à l'étape précédente. Le paramètre `awsvpcConfiguration` est obligatoire, car l'exemple de définition de tâche utilise le mode réseau `awsvpc`.

Lorsque vous créez le service ECS, spécifiez Fargate et la version de plateforme `LATEST` qui prend en charge Service Connect. Les `securityGroups` et les `subnets` doivent appartenir à un VPC répondant aux exigences requises pour utiliser Amazon ECS. Vous pouvez obtenir les ID du groupe de sécurité et du sous-réseau à partir de la console Amazon VPC.

Ce service configure Service Connect en ajoutant le paramètre `serviceConnectConfiguration`. L'espace de noms n'est pas obligatoire car un espace de noms par défaut est configuré pour le cluster. Les applications clientes exécutées dans ECS dans l'espace de noms se connectent à ce service en utilisant le `portName` et le port dans les `clientAliases`. Par exemple, ce service est accessible via `http://nginx:80/`, car nginx fournit une page de bienvenue à l'emplacement racine `/`. Les applications externes qui ne s'exécutent pas dans Amazon ECS ou qui ne se trouvent pas dans le même espace de noms peuvent accéder à cette application via le proxy Service Connect en utilisant l'adresse IP de la tâche et le numéro de port figurant dans la

définition de la tâche. Pour votre configuration `tls`, ajoutez l'`arn` du certificat pour votre `awsPcaAuthorityArn`, votre `kmsKey` et `roleArn` de votre rôle IAM.

Ce service utilise un `logConfiguration` pour envoyer la sortie du proxy Service Connect depuis `stdout` et `stderr` vers Amazon CloudWatch Journaux. Ce rôle d'exécution de tâches ne dispose pas des autorisations supplémentaires requises pour que CloudWatch Groupe de journaux. Créez le groupe de journaux dans CloudWatch Journaux à l'aide du Console de gestion AWS ou AWS CLI. Nous vous recommandons de créer ce groupe de journaux et de stocker les journaux du proxy dans CloudWatch Journaux. Si vous ne souhaitez pas envoyer les journaux du proxy à CloudWatch Journaux que vous pouvez supprimer `logConfiguration`.

```
{
    "cluster": "tutorial",
    "deploymentConfiguration": {
        "maximumPercent": 200,
        "minimumHealthyPercent": 0
    },
    "deploymentController": {
        "type": "ECS"
    },
    "desiredCount": 1,
    "enableECSManagedTags": true,
    "enableExecuteCommand": true,
    "launchType": "FARGATE",
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [
                "sg-EXAMPLE"
            ],
            "subnets": [
                "subnet-EXAMPLE",
                "subnet-EXAMPLE",
                "subnet-EXAMPLE"
            ]
           }
    },
    "platformVersion": "LATEST",
    "propagateTags": "SERVICE",
    "serviceName": "service-connect-nginx-service",
```

```
    "serviceConnectConfiguration": {
        "enabled": true,
        "services": [
            {
                "portName": "nginx",
                "clientAliases": [
                    {
                        "port": 80
                    }
                ],
                "tls": {
                   "issuerCertificateAuthority": {
                      "awsPcaAuthorityArn": "certificateArn"
                   },
                   "kmsKey": "kmsKey",
                   "roleArn": "iamRoleArn"
                }
            }
        ],
        "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
                "awslogs-group": "/ecs/service-connect-proxy",
                "awslogs-region": "region",
                "awslogs-stream-prefix": "service-connect-proxy"
            }
        }
    },
    "taskDefinition": "service-connect-nginx"
}
```

b. Créez un service à l'aide du fichier `service-connect-nginx-service.json` :

```
aws ecs create-service --cluster tutorial --cli-input-json file://service-
connect-nginx-service.json
```

Sortie :

```
{
    "service": {
```

```
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/tutorial/
service-connect-nginx-service",
        "serviceName": "service-connect-nginx-service",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/tutorial",
        "loadBalancers": [],
        "serviceRegistries": [],
        "status": "ACTIVE",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "FARGATE",
        "platformVersion": "LATEST",
        "platformFamily": "Linux",
        "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-definition/
service-connect-nginx:1",
        "deploymentConfiguration": {
            "deploymentCircuitBreaker": {
                "enable": false,
                "rollback": false
            },
            "maximumPercent": 200,
            "minimumHealthyPercent": 0
        },
        "deployments": [
            {
                "id": "ecs-svc/3763308422771520962",
                "status": "PRIMARY",
                "taskDefinition": "arn:aws:ecs:us-west-2:123456789012:task-
definition/service-connect-nginx:1",
                "desiredCount": 1,
                "pendingCount": 0,
                "runningCount": 0,
                "failedTasks": 0,
                "createdAt": 1661210032.602,
                "updatedAt": 1661210032.602,
                "launchType": "FARGATE",
                "platformVersion": "1.4.0",
                "platformFamily": "Linux",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "assignPublicIp": "ENABLED",
                        "securityGroups": [
                            "sg-EXAMPLE"
                        ],
```

```
                        "subnets": [
                            "subnet-EXAMPLEf",
                            "subnet-EXAMPLE",
                            "subnet-EXAMPLE"
                        ]
                    }
                },
                "rolloutState": "IN_PROGRESS",
                "rolloutStateReason": "ECS deployment ecs-
svc/3763308422771520962 in progress.",
                "failedLaunchTaskCount": 0,
                "replacedTaskCount": 0,
                "serviceConnectConfiguration": {
                    "enabled": true,
                    "namespace": "service-connect",
                    "services": [
                        {
                            "portName": "nginx",
                            "clientAliases": [
                                {
                                    "port": 80
                                }
                            ]
                        }
                    ],
                    "logConfiguration": {
                        "logDriver": "awslogs",
                        "options": {
                            "awslogs-group": "/ecs/service-connect-proxy",
                            "awslogs-region": "us-west-2",
                            "awslogs-stream-prefix": "service-connect-proxy"
                        },
                        "secretOptions": []
                    }
                },
                "serviceConnectResources": [
                    {
                        "discoveryName": "nginx",
                        "discoveryArn": "arn:aws:servicediscovery:us-
west-2:123456789012:service/srv-EXAMPLE"
                    }
                ]
            }
        ],
```

```
        "roleArn": "arn:aws:iam::123456789012:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
        "version": 0,
        "events": [],
        "createdAt": 1661210032.602,
        "placementConstraints": [],
        "placementStrategy": [],
        "networkConfiguration": {
            "awsvpcConfiguration": {
                "assignPublicIp": "ENABLED",
                "securityGroups": [
                    "sg-EXAMPLE"
                ],
                "subnets": [
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE",
                    "subnet-EXAMPLE"
                ]
            }
        },
        "schedulingStrategy": "REPLICA",
        "enableECSManagedTags": true,
        "propagateTags": "SERVICE",
        "enableExecuteCommand": true
    }
}
```

La `serviceConnectConfiguration` que vous avez fournie apparaît dans le premier déploiement de la sortie. Lorsque vous apportez des modifications au service ECS de manière à modifier des tâches, un nouveau déploiement est créé par Amazon ECS.

Étape 3 : Vérifier que vous pouvez vous connecter

Pour vérifier que Service Connect est configuré et fonctionne, procédez comme suit pour vous connecter au service Web à partir d'une application externe. Consultez ensuite les mesures supplémentaires dans CloudWatch créé par le proxy Service Connect.

Pour vous connecter au service Web à partir d'une application externe

- Connectez-vous à l'adresse IP de la tâche et au port du conteneur à l'aide de l'adresse IP de la tâche

Utilisez le AWS CLI pour obtenir l'ID de tâche, à l'aide du `aws ecs list-tasks --cluster tutorial`.

Si vos sous-réseaux et votre groupe de sécurité autorisent le trafic depuis l'Internet public sur le port défini par la tâche, vous pouvez vous connecter à l'adresse IP publique depuis votre ordinateur. L'adresse IP publique n'est cependant pas disponible sur `describe-tasks', donc les étapes consistent à accéder à Amazon EC2 Console de gestion AWS ou AWS CLI pour obtenir les détails de l'interface réseau élastique.

Dans cet exemple, une instance Amazon EC2 dans le même VPC utilise l'adresse IP privée de la tâche. L'application est nginx, mais l'en-tête `server: envoy` indique que le proxy Service Connect est utilisé. Le proxy Service Connect écoute sur le port du conteneur depuis la définition de la tâche.

```
$ curl -v 10.0.19.50:80/
*   Trying 10.0.19.50:80...
* Connected to 10.0.19.50 (10.0.19.50) port 80 (#0)
> GET / HTTP/1.1
> Host: 10.0.19.50
> User-Agent: curl/7.79.1
> Accept: */*
>
* Mark bundle as not supporting multiuse
< HTTP/1.1 200 OK
< server: envoy
< date: Tue, 23 Aug 2022 03:53:06 GMT
< content-type: text/html
< content-length: 612
< last-modified: Tue, 16 Apr 2019 13:08:19 GMT
< etag: "5cb5d3c3-264"
< accept-ranges: bytes
< x-envoy-upstream-service-time: 0
<
<!DOCTYPE html>
<html>
<head>
<title>Welcome to nginx!</title>
<style>
    body {
        width: 35em;
```

```
        margin: 0 auto;
        font-family: Tahoma, Verdana, Arial, sans-serif;
    }
</style>
</head>
<body>
<h1>Welcome to nginx!</h1>
<p>If you see this page, the nginx web server is successfully installed and
working. Further configuration is required.</p>

<p>For online documentation and support please refer to
<a href="http://nginx.org/">nginx.org</a>.<br/>
Commercial support is available at
<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>
</body>
</html>
```

Pour consulter les métriques de Service Connect

Le proxy Service Connect crée des métriques d'application (connexion HTTP, HTTP2, gRPC ou TCP) dans CloudWatch métriques. Lorsque vous utilisez le CloudWatch console, voir les dimensions métriques supplémentaires de DiscoveryName, (DiscoveryName, ServiceName, ClusterName), TargetDiscoveryName, et (TargetDiscoveryName, ServiceName, ClusterName) sous l'espace de noms Amazon ECS. Pour plus d'informations sur ces mesures et leurs dimensions, voir Afficher les mesures disponibles en Amazonie CloudWatch Guide de l'utilisateur des journaux.

Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS

Vous pouvez éventuellement configurer votre Amazon ECS service afin d'utiliser la découverte de service Amazon ECS. Utilisations de la découverte de services AWS Cloud Map Actions d'API pour gérer les espaces de noms HTTP et DNS pour vos services Amazon ECS. Pour plus d'informations, consultez Présentation de AWS Cloud Map dans le Manuel du développeur AWS Cloud Map .

La découverte de services est disponible dans les versions suivantes AWS Régions :

Nom de la région	Région
USA Est (Virginie du Nord)	us-east-1
USA Est (Ohio)	us-east-2
USA Ouest (Californie du Nord)	us-west-1
USA Ouest (Oregon)	us-west-2
Afrique (Le Cap)	af-south-1
Asie-Pacifique (Hong Kong)	ap-east-1
Asie-Pacifique (Taipei)	ap-east-2
Asie-Pacifique (Mumbai)	ap-south-1
Asie-Pacifique (Hyderabad)	ap-south-2
Asie-Pacifique (Tokyo)	ap-northeast-1
Asie-Pacifique (Séoul)	ap-northeast-2
Asie-Pacifique (Osaka)	ap-northeast-3
Asie-Pacifique (Singapour)	ap-southeast-1
Asie-Pacifique (Sydney)	ap-southeast-2
Asie-Pacifique (Jakarta)	ap-southeast-3
Asie-Pacifique (Melbourne)	ap-southeast-4
Asie-Pacifique (Malaisie)	ap-southeast-5
Asie-Pacifique (Nouvelle-Zélande)	ap-southeast-6
Asie-Pacifique (Thaïlande)	ap-southeast-7
Canada (Centre)	ca-central-1

Nom de la région	Région
Canada-Ouest (Calgary)	ca-west-1
Chine (Beijing)	cn-north-1
China (Ningxia)	cn-northwest-1
Europe (Francfort)	eu-central-1
Europe (Zurich)	eu-central-2
Europe (Irlande)	eu-west-1
Europe (Londres)	eu-west-2
Europe (Paris)	eu-west-3
Europe (Milan)	eu-south-1
Europe (Stockholm)	eu-north-1
Israël (Tel Aviv)	il-central-1
Europe (Espagne)	eu-south-2
Moyen-Orient (EAU)	me-central-1
Mexique (Centre)	mx-central-1
Moyen-Orient (Bahreïn)	me-south-1
Amérique du Sud (São Paulo)	sa-east-1
AWS GovCloud (US-East)	us-gov-east-1
AWS GovCloud (US-West)	us-gov-west-1

Concepts associés à la découverte de service

La découverte de service se compose des éléments suivants :

- Espace de noms de découverte de services : groupe logique de services de découverte de services qui partagent le même nom de domaine, tel que `example.com`, vers lequel vous souhaitez acheminer le trafic. Vous pouvez créer un espace de noms avec un appel à la commande `aws servicediscovery create-private-dns-namespace` ou dans la console Amazon ECS. Vous pouvez utiliser la commande `aws servicediscovery list-namespaces` pour afficher les informations récapitulatives concernant les espaces de noms créés par le compte actuel. Pour plus d'informations sur les commandes de découverte de services, consultez `create-private-dns-namespace` et `list-namespaces` dans le AWS Cloud Map (découverte de services) AWS CLI Guide de référence.
- Service de découverte de service : présent dans l'espace de noms de la découverte de service, il se compose du nom du service et de la configuration DNS correspondant à l'espace de noms. Il fournit les principaux composants suivants :
  - Registre des services: vous permet de rechercher un service via DNS ou AWS Cloud Map Actions d'API et récupération d'un ou de plusieurs points de terminaison disponibles qui peuvent être utilisés pour se connecter au service.
- Instance de découverte de service : existe dans le service de découverte de service et comprend les attributs associés à chaque service Amazon ECS service dans le répertoire de services.
  - Attributs de l'instance : les métadonnées suivantes sont ajoutées en tant qu'attributs personnalisés pour chaque service Amazon ECS service qui est configuré pour utiliser la découverte de service :
    - **`AWS_INSTANCE_IPV4`** — Pour un `A` record, l'adresse IPv4 que Route 53 renvoie en réponse aux requêtes DNS et AWS Cloud Map renvoie lors de la découverte des détails de l'instance, par exemple, `192.0.2.44`.
    - **`AWS_INSTANCE_IPV6`**— Pour un `AAAA` record, l'adresse IPv6 renvoyée par Route 53 en réponse à des requêtes DNS et AWS Cloud Map renvoie lors de la découverte des détails de l'instance, par exemple, `2001:0db8:85a3:0000:0000:abcd:0001:2345`. `AWS_INSTANCE_IPv4` et `AWS_INSTANCE_IPv6` sont ajoutés pour les services Amazon ECS à double pile. Uniquement `AWS_INSTANCE_IPv6` est ajouté pour Amazon ECS IPv6-only services.
    - **`AWS_INSTANCE_PORT`** – La valeur de port associées au service de découverte de service.
    - **`AVAILABILITY_ZONE`** – La zone de disponibilité dans laquelle la tâche a été lancée. Pour les tâches qui utilisent EC2, il s’agit de la zone de disponibilité dans laquelle l’instance de conteneur existe. Pour les tâches qui utilisent Fargate, il s’agit de la zone de disponibilité dans laquelle l’interface réseau Elastic existe.
    - **`REGION`** – La région dans laquelle la tâche existe.

- **ECS_SERVICE_NAME** – Le nom du service Amazon ECS service auquel la tâche appartient.
- **ECS_CLUSTER_NAME** – Le nom du cluster Amazon ECS auquel la tâche appartient.
- **EC2_INSTANCE_ID** – L'ID de l'instance de conteneur sur laquelle la tâche a été placée. Cet attribut personnalisé n'est pas ajouté si la tâche utilise Fargate.
- **ECS_TASK_DEFINITION_FAMILY** – La famille de définition de tâche que la tâche utilise.
- **ECS_TASK_SET_EXTERNAL_ID** – Si un ensemble de tâches est créé pour un déploiement externe et est associé à un registre de découverte de service, alors l'attribut `ECS_TASK_SET_EXTERNAL_ID` contiendra l'ID externe de l'ensemble de tâches.

- Surveillances de l'état Amazon ECS : Amazon ECS permet d'effectuer des surveillances d'état régulières au niveau du conteneur. Si un point de terminaison échoue à la surveillance de l'état, il est supprimé du routage DNS et marqué comme défectueux.

Remarques relatives à la découverte de service

Les informations suivantes doivent être prises en compte lors de l'utilisation de la découverte de service :

- La découverte de service est prise en charge pour les tâches sur Fargate qui utilisent la version de plateforme 1.1.0 ou supérieure. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.
- Les services configurés pour utiliser la découverte de service sont limités à 1 000 tâches par service. Cela est dû à un quota de service Route 53.
- Le flux de travail créer un service dans la console Amazon ECS prend uniquement en charge l'enregistrement des services dans des espaces de noms DNS privés. Quand un AWS Cloud Map un espace de noms DNS privé est créé, une zone hébergée privée Route 53 sera créée automatiquement.
- Les attributs DNS du VPC doivent être configurés pour une résolution DNS réussie. Pour plus d'informations sur la configuration des attributs, consultez Attributs DNS dans votre VPC dans le Guide de l'utilisateur Amazon VPC.
- Amazon ECS ne prend pas en charge l'enregistrement de services dans un environnement partagé AWS Cloud Map espaces de noms.
- Les registres DNS créés pour un service de découverte de service sont toujours enregistrés avec l'adresse IP privée de la tâche et non pas l'adresse IP publique, même lorsque des espaces de noms publics sont utilisés.

- La découverte de service exige que les tâches spécifient le mode réseau `awsvpc`, `bridge` ou `host` (`none` n'est pas pris en charge).
- Si la définition de tâche du service utilise le mode réseau `awsvpc`, vous pouvez créer n'importe quelle combinaison d'enregistrements `A` ou `SRV` pour chaque tâche de service. Si vous utilisez des enregistrements `SRV`, un port est requis. Vous pouvez également créer des enregistrements `AAAA` si le service utilise des sous-réseaux à double pile. Si le service utilise IPv6-only sous-réseaux, vous ne pouvez pas créer `A` records.
- Si la définition de tâche du service utilise le mode réseau `bridge` ou `host`, un registre SRV est le seul type de registre DNS pris en charge. Créez un registre SRV pour chaque tâche de service. Le registre SRV doit spécifier une combinaison de nom de conteneur et de port de conteneur à partir de la définition de tâche.
- Vous pouvez interroger les registres DNS pour un service de découverte de service au sein de votre VPC. Ces enregistrements utilisent le format suivant : `<service-discovery-service-name>.<service-discovery-namespace>`.
- Lors de l'exécution d'une requête DNS sur le nom du service, les enregistrements `A` et `AAAA` renvoient un ensemble d'adresses IP correspondant à vos tâches. Les enregistrements `SRV` renvoient un ensemble d'adresses IP et de ports pour chaque tâche.
- Si vous disposez de huit registres sains ou moins, Route 53 répond à toutes les requêtes DNS par tous les registres sains.
- Lorsque tous les registres sont non sains, Route 53 répond à toutes les requêtes DNS par jusqu'à huit registres non sains.
- Vous pouvez configurer la découverte de service pour un service situé derrière un équilibreur de charge, mais le trafic de découverte de service est toujours acheminé vers la tâche et non pas vers l'équilibreur de charge.
- La découverte de service ne prend pas en charge l'utilisation des équilibreurs de charge Classic Load Balancer.
- Nous vous recommandons d'utiliser les surveillances de l'état au niveau des conteneurs gérés par Amazon ECS pour votre service de découverte de service.
  - HealthCheckCustomConfig—Amazon ECS gère les bilans de santé en votre nom. Amazon ECS utilise les informations obtenues par le conteneur et les surveillances de l'état, et l'état de votre tâche, afin de mettre à jour l'état avec AWS Cloud Map. Cette configuration est spécifiée à l'aide du paramètre `--health-check-custom-config` au moment de la création de votre service de découverte de service. Pour plus d'informations, consultez HealthCheckCustomConfig dans la Référence d'API AWS Cloud Map .

- Le AWS Cloud Map les ressources créées lors de l'utilisation de la découverte de services doivent être nettoyées manuellement.
- Les tâches et les instances sont enregistrées comme `UNHEALTHY` jusqu'à ce que la surveillance de l'état du conteneur renvoie une valeur. Si les surveillances de l'état réussissent, le statut est mis à jour à `HEALTHY`. Si la surveillance de l'état du conteneur échoue, l'enregistrement de l'instance de découverte de service est annulé.

Tarification de la découverte de service

Les clients utilisant la découverte de service Amazon ECS service sont facturés pour les ressources Route 53 et les opérations d'API de découverte AWS Cloud Map . Le montant facturé englobe les coûts associés à la création des zones hébergées Route 53 et des requêtes sur le registre de services. Pour plus d'informations, consultez Tarification AWS Cloud Map dans le Guide du développeur AWS Cloud Map .

Amazon ECS effectue des contrôles de santé au niveau des conteneurs et les expose à AWS Cloud Map opérations d'API de contrôle de santé personnalisées. Ceci est actuellement mis à la disposition des clients sans frais supplémentaires. Si vous configurez des surveillances de l'état du réseau supplémentaires pour les tâches exposées publiquement, vous êtes facturé pour ces surveillances de l'état.

Création d'un service Amazon ECS qui utilise la découverte de service

Découvrez comment créer un service contenant une tâche Fargate qui utilise la découverte de service avec l' AWS CLI.

Pour une liste des Régions AWS qui prennent en charge la découverte de services, voir Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.

Pour de plus amples informations sur les Régions qui prennent en charge Fargate, veuillez consulter the section called "AWS Régions de Fargate".

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS depuis AWS CLI, des kits de développement logiciel et l'API Amazon ECS sur IPv4 et IPv6. Pour plus d'informations, consultez Utilisation des points de terminaison à double pile Amazon ECS.

Conditions préalables

Avant de commencer ce didacticiel, vérifiez que vous respectez les conditions requises suivantes :

- La dernière version du AWS CLI est installé et configuré. Pour plus d'informations, consultez la section Installation ou mise à jour de la version la plus récente de l' AWS CLI.
- Les étapes décrites dans Configurer l'utilisation d'Amazon ECS sont terminées.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.
- Vous avez créé au moins un VPC et un groupe de sécurité. Pour plus d'informations, consultez the section called "Créer un Virtual Private Cloud".

Étape 1 : créer les ressources Service Discovery dans AWS Cloud Map

Suivez ces étapes suivantes pour créer votre espace de noms de découverte de service et votre service de découverte de service :

1. Créez un espace de noms de découverte de service Cloud Map privé. Cet exemple crée un espace de noms appelé `tutorial`. Remplacer *vpc-abcd1234* avec l'identifiant de l'un de vos VPC existants.

   ```
   aws servicediscovery create-private-dns-namespace \
         --name tutorial \
         --vpc vpc-abcd1234
   ```

   La sortie de cette commande est la suivante.

   ```
   {
       "OperationId": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e"
   }
   ```

2. À l'aide du paramètre `OperationId` obtenu à partir de la sortie de l'étape précédente, vérifiez que l'espace de noms privés a bien été créé. Notez l'ID de l'espace de noms car vous l'utiliserez dans les commandes suivantes.

   ```
   aws servicediscovery get-operation \
         --operation-id h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e
   ```

   La sortie est la suivante.

```
{
    "Operation": {
        "Id": "h2qe3s6dxftvvt7riu6lfy2f6c3jlhf4-je6chs2e",
        "Type": "CREATE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1519777852.502,
        "UpdateDate": 1519777856.086,
        "Targets": {
           "NAMESPACE": "ns-uejictsjen2i4eeg"
        }
    }
}
```

3. En utilisant l'ID NAMESPACE indiqué dans la sortie de l'étape précédente, créez un service de découverte de service. Cet exemple crée un service nommé `myapplication`. Notez l'ID du service et l'ARN car vous les utiliserez dans les commandes suivantes.

```
aws servicediscovery create-service \
      --name myapplication \
      --dns-config "NamespaceId="ns-
uejictsjen2i4eeg",DnsRecords=[{Type="A",TTL="300"}]" \
      --health-check-custom-config FailureThreshold=1
```

La sortie est la suivante.

```
{
    "Service": {
       "Id": "srv-utcrh6wavdkggqtk",
        "Arn": "arn:aws:servicediscovery:region:aws_account_id:service/srv-
utcrh6wavdkggqtk",
        "Name": "myapplication",
        "DnsConfig": {
            "NamespaceId": "ns-uejictsjen2i4eeg",
            "DnsRecords": [
                {
                    "Type": "A",
                    "TTL": 300
                }
            ]
        },
        "HealthCheckCustomConfig": {
```

```
            "FailureThreshold": 1
        },
        "CreatorRequestId": "e49a8797-b735-481b-a657-b74d1d6734eb"
    }
}
```

Étape 2 : Créer les ressources Amazon ECS

Suivez ces étapes pour créer votre cluster, votre définition de tâche et votre service Amazon ECS service :

1. Créez un nouveau cluster Amazon ECS. Cet exemple crée un cluster appelé `tutorial`.

```
aws ecs create-cluster \
      --cluster-name tutorial
```

2. Enregistrez une définition de tâche compatible avec Fargate et qui utilise le mode réseau `awsvpc`. Procédez comme suit :

   a. Créez un fichier nommé `fargate-task.json` avec le contenu de la définition de tâche suivante.

```
{
    "family": "tutorial-task-def",
        "networkMode": "awsvpc",
        "containerDefinitions": [
            {
                "name": "sample-app",
                "image": "public.ecr.aws/docker/library/httpd:2.4",
                "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\""
                ]
            }
        ],
        "requiresCompatibilities": [
            "FARGATE"
        ],
        "cpu": "256",
        "memory": "512"
}
```

b. Enregistrez la définition de tâche en utilisant `fargate-task.json`.

```
aws ecs register-task-definition \
      --cli-input-json file://fargate-task.json
```

3. Créez un service ECS en procédant comme suit :

a. Créez un fichier nommé `ecs-service-discovery.json` avec le contenu du service ECS que vous créez. Cet exemple utilise la définition de tâche créée à l'étape précédente. Le paramètre `awsvpcConfiguration` est obligatoire, car l'exemple de définition de tâche utilise le mode réseau `awsvpc`.

Lorsque vous créez le service ECS, spécifiez le type de lancement Fargate et la version de plateforme `LATEST`, qui prend en charge la fonction de découverte de service. Lorsque le service de découverte de service est créé dans AWS Cloud Map , `registryArn` est l'ARN renvoyé. Les `securityGroups` et les `subnets` doivent appartenir au VPC utilisé pour créer l'espace de noms Cloud Map. Vous pouvez obtenir les ID du groupe de sécurité et du sous-réseau à partir de la console Amazon VPC.

```
{
    "cluster": "tutorial",
    "serviceName": "ecs-service-discovery",
    "taskDefinition": "tutorial-task-def",
    "serviceRegistries": [
       {
```

```
            "registryArn":
 "arn:aws:servicediscovery:region:aws_account_id:service/srv-utcrh6wavdkggqtk"
         }
      ],
      "launchType": "FARGATE",
      "platformVersion": "LATEST",
      "networkConfiguration": {
         "awsvpcConfiguration": {
            "assignPublicIp": "ENABLED",
            "securityGroups": [ "sg-abcd1234" ],
            "subnets": [ "subnet-abcd1234" ]
         }
      },
      "desiredCount": 1
}
```

b. Créez votre service ECS à l'aide du `ecs-service-discovery.json`.

```
aws ecs create-service \
      --cli-input-json file://ecs-service-discovery.json
```

Étape 3 : Vérifiez la découverte du service dans AWS Cloud Map

Vous pouvez vérifier que tout est bien créé en interrogeant les informations associées à la fonction de découverte de service. Une fois la découverte des services configurée, vous pouvez soit utiliser AWS Cloud Map Opérations d'API, ou appel `dig` à partir d'une instance au sein de votre VPC. Procédez comme suit :

1. À l'aide de l'ID de service de découverte de service, répertoriez les instances de découverte de service. Notez l'ID de l'instance (en gras) pour le nettoyage des ressources.

   ```
   aws servicediscovery list-instances \
         --service-id srv-utcrh6wavdkggqtk
   ```

   La sortie est la suivante.

   ```
   {
       "Instances": [
           {
               "Id": "16becc26-8558-4af1-9fbd-f81be062a266",
               "Attributes": {
   ```

```
                "AWS_INSTANCE_IPV4": "172.31.87.2"
                "AWS_INSTANCE_PORT": "80",
                "AVAILABILITY_ZONE": "us-east-1a",
                "REGION": "us-east-1",
                "ECS_SERVICE_NAME": "ecs-service-discovery",
                "ECS_CLUSTER_NAME": "tutorial",
                "ECS_TASK_DEFINITION_FAMILY": "tutorial-task-def"
            }
        }
    ]
}
```

2. Utilisez l'espace de noms, le service et les paramètres supplémentaires tels que le nom du cluster ECS pour demander des informations détaillées sur les instances de la fonction de découverte de service.

```
aws servicediscovery discover-instances \
      --namespace-name tutorial \
      --service-name myapplication \
      --query-parameters ECS_CLUSTER_NAME=tutorial
```

3. Vous pouvez interroger les registres DNS créés dans la zone hébergée Route 53 de votre service de découverte de service à l'aide des commandes AWS CLI suivantes :

   a. À l'aide de l'ID de l'espace de noms, obtenez les informations relatives à l'espace de noms, lesquelles incluent l'ID de la zone hébergée Route 53.

   ```
   aws servicediscovery \
         get-namespace --id ns-uejictsjen2i4eeg
   ```

   La sortie est la suivante.

   ```
   {
       "Namespace": {
           "Id": "ns-uejictsjen2i4eeg",
           "Arn": "arn:aws:servicediscovery:region:aws_account_id:namespace/ns-
   uejictsjen2i4eeg",
           "Name": "tutorial",
           "Type": "DNS_PRIVATE",
           "Properties": {
                "DnsProperties": {
                   "HostedZoneId": "Z35JQ4ZFDRYPLV"
   ```

```
            }
        },
        "CreateDate": 1519777852.502,
        "CreatorRequestId": "9049a1d5-25e4-4115-8625-96dbda9a6093"
    }
}
```

b. À l'aide de l'ID de zone hébergée Route 53 de l'étape précédente (voir le texte en gras), obtenez l'enregistrement de ressource défini pour la zone hébergée.

```
aws route53 list-resource-record-sets \
      --hosted-zone-id Z35JQ4ZFDRYPLV
```

4. Vous pouvez également interroger le DNS à partir d'une instance de votre VPC à l'aide de `dig`.

```
dig +short myapplication.tutorial
```

Étape 4 : nettoyer

Une fois que vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées. Procédez comme suit :

1. Annulez l'enregistrement des instances de service de découverte de service à l'aide de l'ID de service et de l'ID de l'instance que vous avez notés précédemment.

```
aws servicediscovery deregister-instance \
      --service-id srv-utcrh6wavdkggqtk \
      --instance-id 16becc26-8558-4af1-9fbd-f81be062a266
```

La sortie est la suivante.

```
{
    "OperationId": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv"
}
```

2. À l'aide du `OperationId` indiqué dans la sortie de l'étape précédente, vérifiez que l'inscription des instances du service de découverte de service ont été correctement annulées.

```
aws servicediscovery get-operation \
      --operation-id xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv
```

```
{
  "Operation": {
        "Id": "xhu73bsertlyffhm3faqi7kumsmx274n-jh0zimzv",
        "Type": "DEREGISTER_INSTANCE",
        "Status": "SUCCESS",
        "CreateDate": 1525984073.707,
        "UpdateDate": 1525984076.426,
        "Targets": {
            "INSTANCE": "16becc26-8558-4af1-9fbd-f81be062a266",
            "ROUTE_53_CHANGE_ID": "C5NSRG1J4I1FH",
            "SERVICE": "srv-utcrh6wavdkggqtk"
        }
    }
}
```

3. Supprimez le service de découverte de service en utilisant l'ID du service.

```
aws servicediscovery delete-service \
      --id srv-utcrh6wavdkggqtk
```

4. Supprimer l'espace de noms de découverte de service en utilisant l'ID de l'espace de noms.

```
aws servicediscovery delete-namespace \
      --id ns-uejictsjen2i4eeg
```

La sortie est la suivante.

```
{
    "OperationId": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj"
}
```

5. À l'aide du `OperationId` indiqué dans la sortie de l'étape précédente, vérifiez que l'espace de noms de la découverte de service a été correctement supprimé.

```
aws servicediscovery get-operation \
      --operation-id c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj
```

La sortie est la suivante.

```
{
```

```
    "Operation": {
        "Id": "c3ncqglftesw4ibgj5baz6ktaoh6cg4t-jh0ztysj",
        "Type": "DELETE_NAMESPACE",
        "Status": "SUCCESS",
        "CreateDate": 1525984602.211,
        "UpdateDate": 1525984602.558,
        "Targets": {
            "NAMESPACE": "ns-rymlehshst7hhukh",
            "ROUTE_53_CHANGE_ID": "CJP2A2M86XW3O"
        }
    }
}
```

6. Définissez le nombre souhaité pour le service Amazon ECS sur `0`. Vous devez effectuer cette étape pour supprimer le service à l'étape suivante.

```
aws ecs update-service \
      --cluster tutorial \
      --service ecs-service-discovery \
      --desired-count 0
```

7. Supprimez l'Amazon ECS service.

```
aws ecs delete-service \
      --cluster tutorial \
      --service ecs-service-discovery
```

8. Supprimez le cluster Amazon ECS.

```
aws ecs delete-cluster \
      --cluster tutorial
```

Utilisez Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS

Amazon VPC Lattice est un service de mise en réseau d'applications entièrement géré qui permet aux clients Amazon ECS d'observer, de sécuriser et de surveiller les applications créées sur AWS des services informatiques, des VPC et des comptes, sans aucune modification de code.

VPC Lattice utilise des groupes cibles, qui sont un ensemble de ressources de calcul. Ces cibles exécutent votre application ou votre service et peuvent être des instances Amazon EC2, des

adresses IP, des fonctions Lambda et des équilibreurs de charge d'application. En associant leurs services Amazon ECS à un groupe cible VPC Lattice, les clients peuvent désormais activer les tâches Amazon ECS en tant que cibles IP dans VPC Lattice. Amazon ECS enregistre automatiquement les tâches auprès du groupe cible VPC Lattice lorsque les tâches du service enregistré sont lancées.

> **Note**
>
> Lorsque vous utilisez cinq configurations VPC Lattice, votre temps de déploiement peut être légèrement plus long que lorsque vous utilisez moins de configurations.

Une règle d'écoute est utilisée pour transférer le trafic vers un groupe cible spécifié lorsque les conditions sont remplies. Un écouteur vérifie les requêtes de connexion à l'aide du protocole sur le port que vous avez configuré. Un service achemine les requêtes vers ses cibles enregistrées en fonction des règles que vous définissez lors de la configuration de votre écouteur.

Amazon ECS remplace également automatiquement une tâche si elle devient défectueuse selon les surveillances de l'état de VPC Lattice. Une fois associés à VPC Lattice, les clients Amazon ECS peuvent également tirer parti des nombreuses autres fonctionnalités de connectivité, de sécurité et d'observabilité entre les calculs de VPC Lattice, telles que la connexion à des services via des clusters, des VPC et des comptes avec AWS Resource Access Manager, l'intégration IAM pour l'autorisation et l'authentification, et des fonctionnalités avancées de gestion du trafic.

Les avantages de VPC Lattice pour les clients Amazon ECS sont les suivants.

- Productivité accrue des développeurs : VPC Lattice améliore la productivité des développeurs en vous permettant de vous concentrer sur la création de fonctionnalités, tandis que VPC Lattice gère les défis liés au réseau, à la sécurité et à l'observabilité de manière uniforme sur toutes les plateformes de calcul.
- Meilleure posture de sécurité : VPC Lattice permet à vos développeurs d'authentifier et de sécuriser facilement les communications entre les applications et les plateformes informatiques, d'appliquer le chiffrement en transit et de mettre en place des contrôles d'accès granulaires grâce aux politiques d'authentification VPC Lattice. Cela vous permet d'adopter une posture de sécurité renforcée qui répond aux principales exigences réglementaires et de conformité du secteur.
- Capacité de mise à l'échelle et résilience améliorées des applications : VPC Lattice vous permet de créer un réseau d'applications déployées avec des fonctionnalités telles que le routage, l'authentification, l'autorisation et la surveillance basés sur les chemins, les en-têtes et les

méthodes. Ces avantages sont fournis sans frais supplémentaires en termes de ressources sur les charges de travail et peuvent prendre en charge des déploiements multiclusters générant des millions de requêtes par seconde sans ajouter de latence significative.

- Flexibilité de déploiement grâce à une infrastructure hétérogène : VPC Lattice fournit des fonctionnalités cohérentes sur tous les services de calcul tels qu'Amazon ECS, Fargate, Amazon EC2, Amazon EKS et Lambda. permettant à votre organisation de choisir l'infrastructure adaptée à chaque application.

Comment VPC Lattice fonctionne avec les autres services Amazon ECS

L'utilisation de VPC Lattice avec Amazon ECS peut modifier la façon dont vous utilisez d'autres services Amazon ECS, tandis que d'autres restent inchangés.

Application Load Balancers

Vous n'avez plus besoin de créer un Application Load Balancer spécifique devant être utilisé avec le type de groupe cible Application Load Balancer dans VPC Lattice, qui est ensuite lié au service Amazon ECS. Au lieu de cela, il vous suffit de configurer votre service Amazon ECS avec un groupe cible VPC Lattice. Vous pouvez également choisir d'utiliser Application Load Balancer avec Amazon ECS en même temps.

Déploiements Amazon ECS

Les déploiements continus d'Amazon ECS et Amazon ECS blue/green la stratégie de déploiement (y compris ses configurations linéaires et canariennes à transfert de trafic) fonctionne avec VPC Lattice, et Amazon ECS intègre les tâches dans les services et les supprime en toute sécurité pendant le déploiement. Pour les ressources VPC Lattice requises pour blue/green, déploiements linéaires et Canary, voir Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary. Le blue/green déploiement propulsé par CodeDeploy (les `CODE_DEPLOY` contrôleur de déploiement) n'est pas pris en charge.

Pour en savoir plus sur VPC Lattice, consultez le Guide de l'utilisateur Amazon VPC Lattice.

Créez un service qui utilise VPC Lattice

Vous pouvez utiliser soit le Console de gestion AWS ou le AWS CLI pour créer un service avec VPC Lattice.

Conditions préalables

Avant de commencer ce didacticiel, vérifiez que vous respectez les conditions requises suivantes :

- La dernière version du AWS CLI est installé et configuré. Pour plus d'informations, consultez Installing the AWS Command Line Interface (Installation de).

  > **Note**
  >
  > Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS depuis AWS CLI, des kits de développement logiciel et l'API Amazon ECS sur IPv4 et IPv6. Pour plus d'informations, consultez Utilisation des points de terminaison à double pile Amazon ECS.

- Les étapes décrites dans Configurer l'utilisation d'Amazon ECS sont terminées.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.

Créez un service qui utilise VPC Lattice avec Console de gestion AWS

Procédez comme suit pour créer un service avec VPC Lattice à l'aide de l' Console de gestion AWS.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans la page de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster dans lequel créer le service.
4. Sous l'onglet Services choisissez Create (Créer).

   Si vous n'avez jamais créé de service auparavant, suivez les étapes décrites dans Création d'un service Amazon ECS à l'aide de la console, puis poursuivez ces étapes lorsque vous atteignez la section VPC Lattice.
5. Choisissez Activer le réseau VPC en cochant le bouton.
6. Pour utiliser un rôle existant, pour Rôle d'infrastructure ECS pour Amazon ECS, choisissez-en un que vous avez déjà créé pour l'utiliser lors de la création du groupe cible VPC Lattice. Pour créer un rôle, Créer un rôle d'infrastructure ECS.
7. Choisissez le VPC.

   Le VPC dépend du mode réseau que vous avez sélectionné lors de l'enregistrement de votre définition de tâche. Si vous utilisez le mode `host` ou `network` avec EC2, choisissez votre VPC.

   Pour le mode `awsvpc`, le VPC est automatiquement sélectionné en fonction du VPC que vous avez choisi sous Mise en réseau et ne peut pas être modifié.

8. Sous Groupes cibles, sélectionnez le ou les groupes cibles. Vous devez choisir au moins un groupe cible et pouvez en avoir jusqu'à cinq. Sélectionnez Ajouter un groupe cible pour ajouter d'autres groupes cibles. Choisissez le Nom du port, le Protocole et le Port pour chaque groupe cible que vous avez choisi. Pour supprimer un groupe cible, sélectionnez Supprimer.

   > **Note**
   >
   > - Si vous souhaitez ajouter un groupe cible existant, vous devez utiliser l' AWS CLI. Pour obtenir des instructions sur la façon d'ajouter des groupes cibles à l'aide du AWS CLI, voir enregistrez-cibles dans le AWS Command Line Interface Référence.
   > - Bien qu'un service VPC Lattice puisse avoir plusieurs groupes cibles, chaque groupe cible ne peut être ajouté qu'à un seul service.
   > - Pour créer un service dans IPv6-only configuration, choisissez des groupes cibles avec un type d'adresse IP de `IPv6`.

9. À ce stade, vous accédez à la console VPC Lattice pour poursuivre la configuration. C'est ici que vous incluez vos nouveaux groupes cibles dans l'action par défaut de l'écouteur ou dans les règles d'un service VPC Lattice existant.

   Pour plus d'informations, consultez la section Règles d'écoute pour votre service VPC Lattice.

> **Important**
>
> Vous devez autoriser le préfixe de règle de trafic entrant `vpc-lattice` à accéder à votre groupe de sécurité, sinon les tâches et la surveillance de l'état échoueront.

Créez un service qui utilise VPC Lattice avec AWS CLI

Utilisez le AWS CLI pour créer un service avec VPC Lattice. Remplacez chaque *`user input placeholder`* par vos propres informations.

1. Créez un fichier de configuration de groupe cible. L'exemple suivant est nommé `tg-config.json`

```
{
    "ipAddressType": "IPV4",
```

```
    "port": 443,
    "protocol": "HTTPS",
    "protocolVersion": "HTTP1",
    "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
}
```

2. Utilisez la commande suivante pour créer un groupe cible VPC Lattice.

```
aws vpc-lattice create-target-group \
    --name my-lattice-target-group-ip \
    --type IP \
    --config file://tg-config.json
```

> **Note**
>
> Pour créer un service dans IPv6-only configuration, créez des groupes cibles avec un type d'adresse IP de `IPv6`. Pour plus d'informations, consultez la section create-target-group dans la Référence des commandes AWS CLI .

Exemple de sortie :

```
{
    "arn": "arn:aws:vpc-lattice:us-east-2:123456789012:targetgroup/
tg-0eaa4b9ab4EXAMPLE",
    "config": {
        "healthCheck": {
            "enabled": true,
            "healthCheckIntervalSeconds": 30,
            "healthCheckTimeoutSeconds": 5,
            "healthyThresholdCount": 5,
            "matcher": {
                "httpCode": "200"
            },
            "path": "/",
            "protocol": "HTTPS",
            "protocolVersion": "HTTP1",
            "unhealthyThresholdCount": 2
        },
        "ipAddressType": "IPV4",
        "port": 443,
```

```
        "protocol": "HTTPS",
        "protocolVersion": "HTTP1",
        "vpcIdentifier": "vpc-f1663d9868EXAMPLE"
    },
    "id": "tg-0eaa4b9ab4EXAMPLE",
    "name": "my-lattice-target-group-ip",
    "status": "CREATE_IN_PROGRESS",
    "type": "IP"
}
```

3. Le fichier JSON suivant nommé *ecs-service-vpc-lattice.json* est un exemple utilisé pour associer un service Amazon ECS à un groupe cible VPC Lattice. Le `portName` dans l'exemple suivant est le même que celui que vous avez défini dans votre définition de tâche `portMappings` de la propriété `name` champ.

```
{
    "serviceName": "ecs-service-vpc-lattice",
    "taskDefinition": "ecs-task-def",
        "vpcLatticeConfigurations": [
        {
            "targetGroupArn": "arn:aws:vpc-lattice:us-
west-2:123456789012:targetgroup/tg-0eaa4b9ab4EXAMPLE",
            "portName": "testvpclattice",
            "roleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRoleVpcLattice"
        }
    ],
    "desiredCount": 5,
    "role": "ecsServiceRole"
}
```

Utilisez la commande suivante pour créer un service Amazon ECS et l'associer au groupe cible VPC Lattice à l'aide de l'exemple JSON précédent.

```
aws ecs create-service \
    --cluster clusterName \
    --serviceName ecs-service-vpc-lattice \
    --cli-input-json file://ecs-service-vpc-lattice.json
```

À utiliser blue/green, pour les déploiements linéaires ou Canary avec VPC Lattice, ajoutez un `advancedConfiguration` bloc (le groupe cible alternatif et les règles d'écoute de

production et de test) vers le `vpcLatticeConfigurations` entrée et définissez la stratégie de déploiement dans `deploymentConfiguration`. Pour plus d'informations, consultez Ressources VPC Lattice pour blue/greendéploiements linéaires et Canary.

> **Note**
>
> VPC Lattice n'est pas pris en charge sur Amazon ECS Anywhere.

Protection de vos tâches Amazon ECS pour éviter qu'elles ne soient interrompues par des événements de réduction horizontale

Vous pouvez utiliser la protection évolutive des tâches Amazon ECS pour protéger vos tâches contre l'interruption d'événements de mise à l'échelle horizontale provenant du service autoscaling ou de déploiements.

Certaines applications nécessitent un mécanisme permettant de protéger les tâches critiques contre toute interruption due à des événements de mise à l'échelle horizontale en période de faible utilisation ou lors de déploiements de services. Par exemple :

- Vous disposez d'une application asynchrone de traitement des files d'attente, telle qu'une tâche de transcodage vidéo, dans laquelle certaines tâches doivent s'exécuter pendant des heures, même lorsque l'utilisation cumulée des services est faible.
- Vous exploitez une application de jeu qui exécute des serveurs de jeu sous forme de tâches Amazon ECS, lesquelles doivent continuer de fonctionner même si tous les utilisateurs se sont déconnectés, afin de réduire la latence de démarrage lors du redémarrage du serveur.
- Lorsque vous déployez une nouvelle version de code, vous avez besoin que des tâches continuent à s'exécuter, car leur retraitement serait coûteux.

Pour empêcher les tâches appartenant à votre service de se terminer lors d'un événement de mise à l'échelle horizontale, définissez l'attribut `ProtectionEnabled` sur `true`. Lorsque vous définissez `ProtectionEnabled` sur true, les tâches sont protégées pendant deux heures par défaut. Vous pouvez ensuite personnaliser la période de protection à l'aide de l'attribut `ExpiresInMinutes`. Vous pouvez protéger vos tâches pendant au moins une minute et jusqu'à un maximum de 2 880 minutes (48 heures). Si vous utilisez AWS CLI, vous pouvez spécifier `--protection-enabled` option.

Une fois qu'une tâche a terminé son travail requis, vous pouvez définir l'attribut `ProtectionEnabled` sur `false`, ce qui permet à la tâche d'être interrompue par des événements ultérieurs de mise à l'échelle horizontale. Si vous utilisez AWS CLI, vous pouvez spécifier `--no-protection-enabled` option.

Mécanismes de protection évolutive des tâches

Vous pouvez définir et obtenir une protection évolutive des tâches à l'aide du point de terminaison de l'agent de conteneur Amazon ECS ou de l'API Amazon ECS.

- Point de terminaison de l'agent de conteneur Amazon ECS

  Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection. Utilisez cette approche pour les charges de travail basées sur les files d'attente ou le traitement des tâches.

  Lorsqu'un conteneur commence à traiter un travail, par exemple en consommant un message SQS, vous pouvez définir l'attribut `ProtectionEnabled` via le chemin `$ECS_AGENT_URI/task-protection/v1/state` du point de terminaison de protection évolutive des tâches depuis le conteneur. Amazon ECS n'interrompra pas cette tâche lors d'événements de mise à l'échelle horizontale. Une fois que votre tâche a terminé son travail, vous pouvez effacer l'attribut `ProtectionEnabled` en utilisant le même point de terminaison, ce qui rend la tâche susceptible d'être interrompue lors d'événements ultérieurs de mise à l'échelle horizontale.

  Pour de plus amples informations sur l'utilisation de point de terminaison de l'agent de conteneur Amazon ECS, consultez la section Point de terminaison de protection évolutive des tâches Amazon ECS.

- API Amazon ECS

  Vous pouvez utiliser l'API Amazon ECS pour définir et récupérer la protection évolutive des tâches si votre application dispose d'un composant qui suit l'état des tâches actives. Utilisez `UpdateTaskProtection` pour marquer une ou plusieurs tâches comme protégées. Utilisez `GetTaskProtection` pour récupérer l'état de protection.

  Un exemple de cette approche serait si votre application héberge des sessions de serveur de jeu comme tâches Amazon ECS. Lorsqu'un utilisateur se connecte à une session sur le serveur (tâche), vous pouvez marquer la tâche comme protégée. Une fois que l'utilisateur se déconnecte, vous pouvez soit retirer la protection spécifique à cette tâche, soit retirer périodiquement la

protection pour des tâches similaires qui ne comportent plus de sessions actives, en fonction de vos besoins en termes de serveurs inactifs.

Pour plus d'informations, voir UpdateTaskProtection et GetTaskProtection dans le Référence de l'API Amazon Elastic Container Service.

Vous pouvez combiner les deux approches. Par exemple, utilisez le point de terminaison de l'agent Amazon ECS pour définir la protection des tâches depuis un conteneur et utilisez l'API Amazon ECS pour retirer la protection des tâches depuis votre service de contrôleur externe.

Considérations

Prenez en compte les points suivants avant d'utiliser la protection évolutive des tâches :

- La protection évolutive des tâches n’est prise en charge que pour les tâches déployées à partir d’un service.
- La protection évolutive des tâches est prise en charge par les tâches déployées à partir d’un service exécuté sur les instances gérées Amazon ECS.
- Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS, car l'agent Amazon ECS possède des mécanismes de nouvelle tentative intégrés et une interface plus simple.
- Vous pouvez réinitialiser la période d'expiration de la protection contre la mise à l'échelle horizontale des tâches en appelant `UpdateTaskProtection` pour une tâche pour laquelle la protection est déjà activée.
- Déterminez le temps nécessaire à une tâche pour effectuer le travail requis et définissez la propriété `expiresInMinutes` en conséquence. Si vous fixez une période d'expiration de la protection plus longue que nécessaire, vous devrez supporter des coûts et serez confronté à des retards dans le déploiement de nouvelles tâches.
- Avec `DEFERRED` nettoyage des révisions du service source, Amazon ECS nettoie les tâches relatives à la révision du service source une fois le déploiement terminé, et essaie pendant deux semaines au maximum. Amazon ECS ne peut pas arrêter les tâches dotées d'une protection évolutive. Si les tâches restent protégées pendant plus de deux semaines après le déploiement, Amazon ECS ne peut pas les nettoyer. Pour plus d’informations, consultez Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces.

- La protection contre la mise à l'échelle horizontale des tâches est prise en charge sur l'agent de conteneur Amazon ECS version `1.65.0` ou version ultérieure. Vous pouvez ajouter la prise en charge de cette fonctionnalité sur les instances Amazon EC2 utilisant des versions plus anciennes de l'agent de conteneur Amazon ECS en mettant à jour l'agent à la dernière version. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- Considérations relatives au déploiement :
  - Si le service utilise une mise à jour propagée, de nouvelles tâches seront créées, mais les tâches exécutant une ancienne version ne seront pas terminées avant la désactivation ou l'expiration de `protectionEnabled`. Vous pouvez ajuster le paramètre `maximumPercentage` dans la configuration du déploiement sur une valeur qui permet de créer de nouvelles tâches lorsque les anciennes tâches sont protégées.
  - Si un blue/green la mise à jour est appliquée, le déploiement bleu avec les tâches protégées ne sera pas supprimé si les tâches ont `protectionEnabled`. Le trafic sera redirigé vers les nouvelles tâches qui apparaissent, et les anciennes tâches ne seront pas supprimées que lorsque `protectionEnabled` sera désactivée ou expirera. En fonction du délai d'expiration du CodeDeploy ou CloudFormation mises à jour, le déploiement peut expirer et les anciennes tâches Blue peuvent toujours être présentes.
  - Si vous utilisez CloudFormation, la pile de mise à jour a un délai d'expiration de 3 heures. Par conséquent, si vous configurez la protection de vos tâches pendant plus de 3 heures, CloudFormation le déploiement peut entraîner un échec et un retour en arrière.

    Pendant la durée de protection de vos anciennes tâches, le CloudFormation expositions empilées `UPDATE_IN_PROGRESS`. Si la protection contre la mise à l'échelle horizontale des tâches est supprimée ou expire dans un délai de trois heures, votre déploiement réussira et passera au statut `UPDATE_COMPLETE`. Si le déploiement est bloqué dans l'état `UPDATE_IN_PROGRESS` pendant plus de 3 heures, il échouera et affichera un état `UPDATE_FAILED`, puis reviendra au dernier ensemble de tâches.
  - Amazon ECS envoie des événements de service lorsque des tâches protégées permettent de maintenir un déploiement (continu ou blue/green) d'atteindre l'état stable, afin que vous puissiez prendre des mesures correctives. Lorsque vous essayez de mettre à jour l'état de protection d'une tâche et que vous recevez un message d'erreur `DEPLOYMENT_BLOCKED`, cela signifie que le service possède un nombre de tâches protégées supérieur au nombre de tâches souhaité pour le service. Pour résoudre cette erreur, effectuez l'une des opérations suivantes :
    - Attendez que la protection des tâches en cours expire. Définissez ensuite la protection des tâches.

- Déterminez quelles tâches peuvent être interrompues. Utilisez ensuite `UpdateTaskProtection` avec l'option `protectionEnabled` définie sur `false` pour ces tâches.
- Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

Autorisations IAM requises pour la protection évolutive des tâches

La tâche doit avoir le rôle de tâche Amazon ECS avec les autorisations suivantes :

- `ecs:GetTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `GetTaskProtection`.
- `ecs:UpdateTaskProtection` : permet à l'agent de conteneur Amazon ECS d'appeler `UpdateTaskProtection`.

Point de terminaison de protection évolutive des tâches Amazon ECS

L'agent de conteneur Amazon ECS injecte automatiquement la variable d'environnement `ECS_AGENT_URI` dans les conteneurs des tâches Amazon ECS afin de fournir une méthode permettant d'interagir avec le point de terminaison de l'API de l'agent de conteneur.

Nous vous recommandons d'utiliser le point de terminaison de l'agent de conteneur Amazon ECS pour les tâches qui peuvent déterminer automatiquement le besoin de protection.

Lorsqu'un conteneur commence à traiter une tâche, vous pouvez définir l'attribut `protectionEnabled` à l'aide du chemin d'accès `$ECS_AGENT_URI/task-protection/v1/state` du point de terminaison de protection évolutive des tâches à partir du conteneur.

Utilisez une requête PUT vers cette URI à partir d'un conteneur pour définir la protection évolutive des tâches. Une requête GET vers cette URI renvoie le statut de protection actuel d'une tâche.

Paramètres de requête de protection évolutive des tâches

Vous pouvez définir la protection évolutive des tâches à l'aide du point de terminaison `${ECS_AGENT_URI}/task-protection/v1/state` avec les paramètres de demande suivants.

`ProtectionEnabled`

Spécifiez `true` pour marquer une tâche à protéger. Spécifiez `false` si vous souhaitez supprimer la protection et rendre la tâche éligible à l'arrêt.

Type : Boolean

Obligatoire : oui

`ExpiresInMinutes`

Le nombre de minutes pendant lesquelles la tâche est protégée. Vous pouvez spécifier un minimum de 1 minute et aller jusqu'à 2 880 minutes (48 heures). Pendant cette période, votre tâche ne sera pas interrompue par des événements de mise à l'échelle horizontale provenant de l'autoscaling du service ou de déploiements. Une fois cette période écoulée, le paramètre `protectionEnabled` est défini sur `false`.

Si vous ne spécifiez pas l'heure, la tâche est automatiquement protégée pendant 120 minutes (2 heures).

Type : Integer

Obligatoire : non

Les exemples suivants montrent comment définir la protection des tâches avec des durées différentes.

Exemple de protection d'une tâche avec la durée par défaut

Cet exemple montre comment protéger une tâche dont la durée par défaut est de 2 heures.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true}'
```

Exemple de protection d'une tâche pendant 60 minutes

Cet exemple montre comment protéger une tâche pendant 60 minutes à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":60}'
```

Exemple de protection d'une tâche pendant 24 heures

Cet exemple montre comment protéger une tâche pendant 24 heures à l'aide du paramètre `expiresInMinutes`.

```
curl --request PUT --header 'Content-Type: application/json' ${ECS_AGENT_URI}/task-
protection/v1/state --data '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}'
```

Exemples pour des conteneurs Windows

Pour les conteneurs Windows, vous pouvez utiliser PowerShells `Invoke-RestMethod` applet de commande au lieu de curl. Les exemples suivants montrent PowerShell équivalents des commandes curl précédentes.

Exemple de protection d'une tâche de conteneur Windows avec la durée par défaut

Cet exemple montre comment protéger une tâche avec la période par défaut de 2 heures à l'aide de PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true}' -ContentType 'application/json'
```

Exemple de protection d'une tâche de conteneur Windows pendant 60 minutes

Cet exemple montre comment protéger une tâche pendant 60 minutes à l'aide du `expiresInMinutes` paramètre avec PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":60}' -ContentType 'application/json'
```

Exemple de protection d'une tâche de conteneur Windows pendant 24 heures

Cet exemple montre comment protéger une tâche pendant 24 heures à l'aide du `expiresInMinutes` paramètre avec PowerShell.

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Put -Body
 '{"ProtectionEnabled":true,"ExpiresInMinutes":1440}' -ContentType 'application/json'
```

La requête PUT renvoie la réponse suivante.

```
{
  "protection": {
    "ExpirationDate": "2023-12-20T21:57:44.837Z",
    "ProtectionEnabled": true,
    "TaskArn": "arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
  }
}
```

Paramètres de réponse de protection évolutive des tâches

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de protection contre la mise à l'échelle horizontale des tâches `${ECS_AGENT_URI}/task-protection/v1/state`.

`ExpirationDate`

Période pendant laquelle la protection de la tâche expirera. Si la tâche n’est pas protégée, cette valeur est nulle.

`ProtectionEnabled`

État de protection de la tâche. Si la protection évolutive est activée pour une tâche, la valeur est `true`. Sinon, la valeur est `false`.

`TaskArn`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

L'exemple suivant affiche les détails renvoyés pour une tâche protégée.

```
curl --request GET ${ECS_AGENT_URI}/task-protection/v1/state
```

Pour les conteneurs Windows, utilisez ce qui suit PowerShell commande pour obtenir l'état de protection :

```
Invoke-RestMethod -Uri $env:ECS_AGENT_URI/task-protection/v1/state -Method Get
```

```
{
    "protection":{
        "ExpirationDate":"2023-12-20T21:57:44Z",
        "ProtectionEnabled":true,
```

```
        "TaskArn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0"
    }
}
```

Les informations suivantes sont renvoyées en cas d'échec.

`Arn`

Amazon Resource Name (ARN) complet de la tâche.

`Detail`

Détails relatifs à l'échec.

`Reason`

Raison de l'échec.

L'exemple suivant affiche les détails renvoyés pour une tâche qui n'est pas protégée.

```
{
    "failure":{
        "Arn":"arn:aws:ecs:us-west-2:111122223333:task/1234567890abcdef0",
        "Detail":null,
        "Reason":"TASK_NOT_VALID"
    }
}
```

Les informations suivantes sont renvoyées en cas d'exception.

`requestID`

Le AWS ID de demande pour l'appel d'API Amazon ECS qui entraîne une exception.

`Arn`

Amazon Resource Name (ARN) complet de la tâche ou du service.

`Code`

Code de l’erreur.

`Message`

Message d’erreur.

> **Note**
>
> Si une erreur `RequestError` ou `RequestTimeout` apparaît, il s'agit probablement d'un problème de mise en réseau. Essayez d'utiliser les points de terminaison d'un VPC pour Amazon ECS.

L'exemple suivant affiche les détails renvoyés en cas d'erreur.

```
{
    "requestID":"12345-abc-6789-0123-abc",
    "error":{
        "Arn":"arn:aws:ecs:us-west-2:555555555555:task/my-cluster-
name/1234567890abcdef0",
        "Code":"AccessDeniedException",
        "Message":"User: arn:aws:sts::444455556666:assumed-role/my-ecs-task-
role/1234567890abcdef0 is not authorized to perform: ecs:GetTaskProtection on resource:
 arn:aws:ecs:us-west-2:555555555555:task/test/1234567890abcdef0 because no identity-
based policy allows the ecs:GetTaskProtection action"
    }
}
```

L'erreur suivante s'affiche si l'agent Amazon ECS ne parvient pas à obtenir de réponse du point de terminaison Amazon ECS pour des raisons telles que des problèmes de réseau ou si le plan de contrôle Amazon ECS est en panne.

```
{
  "error": {
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "RequestCanceled",
    "Message": "Timed out calling Amazon ECS Task Protection API"
  }
}
```

L'erreur suivante apparaît lorsque l'agent Amazon ECS reçoit une exception de limitation de la part d'Amazon ECS.

```
{
  "requestID": "12345-abc-6789-0123-abc",
  "error": {
```

```
    "Arn": "arn:aws:ecs:us-west-2:555555555555:task/my-cluster-name/1234567890abcdef0",
    "Code": "ThrottlingException",
    "Message": "Rate exceeded"
  }
}
```

Utilisation de l'injection de pannes avec vos charges de travail Amazon ECS et Fargate

Les clients peuvent utiliser l'injection de pannes avec Amazon ECS sur Amazon EC2 et Fargate pour tester la manière dont leur application répond à certains scénarios de défaillance. Ces tests fournissent des informations que vous pouvez utiliser pour optimiser les performances et la résilience de votre application.

Lorsque l'injection de pannes est activée, l'agent de conteneur Amazon ECS permet aux tâches d'accéder à de nouveaux points de terminaison d'injection de pannes. Vous devez vous inscrire pour utiliser l'injection de pannes en définissant la valeur du paramètre de définition de la tâche `enableFaultInjection` sur `true`. La valeur par défaut est `false`.

```
{
    ...
   "enableFaultInjection": true
}
```

> **Note**
>
> L'injection de pannes ne fonctionne qu'avec les tâches utilisant le mode réseau `awsvpc` ou `host`.
> L'injection de pannes n'est pas disponible sous Windows.

Pour plus d'informations sur la façon d'activer l'injection de défauts dans Console de gestion AWS, voir Création d'une définition de tâche Amazon ECS à l'aide de la console.

Vous devez activer la fonctionnalité pour la tester dans AWS Fault Injection Service. Pour plus d'informations, voir Utilisez le AWS FIS aws:ecs:task actions.

> **Note**
>
> Si vous n'utilisez pas les nouvelles AMI optimisées pour Amazon ECS ou si vous disposez d'une AMI personnalisée, installez les dépendances suivantes :
>
> - `tc`
> - Module noyau `sch_netem`

Points de terminaison d'injection de pannes Amazon ECS

L'agent de conteneur Amazon ECS injecte automatiquement la variable d'environnement `ECS_AGENT_URI` dans les conteneurs des tâches Amazon ECS afin de fournir une méthode permettant d'interagir avec le point de terminaison de l'API de l'agent de conteneur. Chaque point de terminaison inclut un point de terminaison `/start`, `/stop`, et `/status`. Les points de terminaison n'acceptent que les requêtes provenant de tâches qui ont activé l'injection de pannes, et chaque point de terminaison est limité à une requête toutes les cinq secondes par conteneur. Le dépassement de cette limite entraîne une erreur.

> **Note**
>
> La `version 1.88.0+` de l'agent Amazon ECS est requise pour utiliser les points de terminaison d'injection de pannes.

Les trois points de terminaison à utiliser avec l'injection de pannes sont les suivants :

- Point de terminaison du port réseau blackhole
- Point de terminaison de perte de paquets réseau
- Point de terminaison de latence du réseau

Une requête réussie génère un code de réponse `200` avec un message `running` lorsque vous appelez le point de terminaison `/start`, `stopped` pour le point de terminaison `/stop` et `running` ou `not-running` pour le point de terminaison `/status`.

```
{
    "Status": <string>
```

```
}
```

Une requête infructueuse renvoie l'un des codes d'erreur suivants :

- `400` : requête erronée
- `409` : la requête d'injection de pannes est en conflit avec une autre injection panne en cours
- `429` : la requête a été limitée
- `500` : une erreur inattendue s'est produite sur le serveur

```
{
 "Error":  <string message>
}
```

> **Note**
>
> Une seule erreur de latence réseau ou une seule erreur de perte de paquets réseau peut être injectée à la fois. Toute tentative d'injection multiple entraîne le rejet de la requête.

Point de terminaison du port réseau blackhole

Le point de terminaison `{ECS_AGENT_URI}/fault/v1/network-blackhole-port` élimine le trafic entrant ou sortant pour un port et un protocole spécifiques dans l'espace de noms réseau d'une tâche et est compatible avec deux modes :

- awsvpc : les modifications sont appliquées à l'espace de noms du réseau de tâches
- hôte : les modifications sont appliquées à l'instance de conteneur d'espace de noms réseau par défaut

{ECS_AGENT_URI}/fault/v1/réseau-blackhole-port/start

Ce point de terminaison démarre les injections de pannes du port réseau blackhole et possède les paramètres suivants :

Port

Le port spécifié à utiliser pour l'injection de pannes du port blackhole.

Type : Integer

Obligatoire : oui

Protocole

Le port à utiliser pour l'injection de pannes du port blackhole.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Le type de trafic utilisé par l'injection de pannes.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

SourcesToFilter

Un tableau JSON d'adresses IPv4 ou IPv6 ou de blocs d'adresses CIDR protégés contre la panne.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande pour utiliser le `start` point de terminaison (remplacez le *red* valeurs (avec les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/start

Http method:POST

Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress"
```

```
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/réseau-blackhole-port/stop

Ce point de terminaison arrête la panne spécifiée dans la requête. Ce point de terminaison a les paramètres suivants :

Port

Le port concerné par la panne qui doit être arrêtée.

Type : Integer

Obligatoire : oui

Protocole

Le protocole à utiliser pour arrêter la panne.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Le type de trafic utilisé par l'injection de pannes.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

Voici un exemple de demande pour utiliser le `stop` point de terminaison (remplacez le *red* valeurs (avec les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/stop

Http method: POST
```

```
Request payload:
{
    "Port": 1234,
    "Protocol": "tcp|udp",
    "TrafficType": "ingress|egress",
}
```

{ECS_AGENT_URI}/fault/v1/réseau-blackhole-port/status

Ce point de terminaison est utilisé pour vérifier l'état de l'injection de pannes. Ce point de terminaison a les paramètres suivants :

Port

Le port concerné pour vérifier l'état de la panne.

Type : Integer

Obligatoire : oui

Protocole

Le protocole à utiliser pour vérifier l'état de la panne.

Type : String

Valeurs valides : `tcp | udp`

Obligatoire : oui

TrafficType

Le type de trafic utilisé par l'injection de pannes.

Type : String

Valeurs valides : `ingress | egress`

Obligatoire : oui

Voici un exemple de demande pour utiliser le `status` point de terminaison (remplacez le *red* valeurs (avec les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-blackhole-port/status
```

```
Http method: POST

Request payload:
{
   "Port": 1234,
   "Protocol": "tcp|udp",
   "TrafficType": "ingress|egress",
}
```

Point de terminaison de latence du réseau

Le point de terminaison `{ECS_AGENT_URI}/fault/v1/network-latency` ajoute du retard et de l'instabilité à l'interface réseau de la tâche pour le trafic vers une source spécifique. Le point de terminaison est compatible avec deux modes :

- awsvpc : les modifications sont appliquées à l'interface réseau de la tâche
- hôte : les modifications sont appliquées à l'interface réseau par défaut

{ECS_AGENT_URI}/fault/v1/réseau-latency/start

Ce point de terminaison `/start` commence l'injection de pannes de latence du réseau et possède les paramètres suivants :

DelayMilliseconds

Le nombre de millisecondes de délai à ajouter à l'interface réseau à utiliser pour l'injection de pannes.

Type : Integer

Obligatoire : oui

JitterMilliseconds

Le nombre de millisecondes d'instabilité à ajouter à l'interface réseau à utiliser pour l'injection de pannes.

Type : Integer

Obligatoire : oui

Sources

Un tableau JSON d'adresses IPv4 ou IPv6 ou de blocs d'adresses CIDR destinés à être utilisés pour l'injection de pannes.

Type : tableau de chaînes

Obligatoire : oui

SourcesToFilter

Un tableau JSON d'adresses IPv4 ou IPv6 ou de blocs d'adresses CIDR protégés contre la panne. `SourcesToFilter` a la priorité sur `Sources`.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande pour utiliser le `/start` point de terminaison (remplacez le *red* valeurs (avec les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/start

Http method: POST

Request payload:
{
    "DelayMilliseconds": 123,
    "JitterMilliseconds": 123,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/réseau-latency/stop et /status

Le point de terminaison `{ECS_AGENT_URI}/fault/v1/network-latency/stop` arrête le défaut et `{ECS_AGENT_URI}/fault/v1/network-latency/status` vérifie l'état du défaut.

Voici deux exemples de requêtes d'utilisation des points de terminaison `/stop` et `/status`. Les deux utilisent la méthode `POST HTTP`.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/stop
```

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-latency/status
```

Point de terminaison de perte de paquets réseau

Le point de terminaison `{ECS_AGENT_URI}/fault/v1/network-packet-loss` ajoute la perte de paquets à l'interface réseau donnée. Ce point de terminaison est compatible avec deux modes :

- awsvpc : les modifications sont appliquées à l'interface réseau de la tâche
- hôte : les modifications sont appliquées à l'interface réseau par défaut

{ECS_AGENT_URI}/fault/v1/packet-réseau-loss/start

Ce point de terminaison `/start` démarre l'injection de pannes de perte de paquets réseau et possède les paramètres suivants :

LossPercent

Le pourcentage de perte de paquets

Type : Integer

Obligatoire : oui

Sources

Un tableau JSON d'adresses IPv4 ou IPv6 ou de blocs d'adresses CIDR à utiliser pour les tests d'injection de pannes.

Type : tableau de chaînes

Obligatoire : oui

SourcesToFilter

Un tableau JSON d'adresses IPv4 ou IPv6 ou de blocs d'adresses CIDR protégés contre la panne. `SourcesToFilter` a la priorité sur `Sources`.

Type : tableau de chaînes

Obligatoire : non

Voici un exemple de demande pour utiliser le `start` point de terminaison (remplacez le *`red`* valeurs (avec les vôtres) :

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/start

Http method: POST

{
    "LossPercent": 6,
    "Sources": ["${IP1}", "${IP2}", ...],
    "SourcesToFilter": ["${IP1}", "${IP2}", ...],
}
```

{ECS_AGENT_URI}/fault/v1/packet-réseau-loss/stop et /status

Le point de terminaison `{ECS_AGENT_URI}/fault/v1/network-packet-loss/stop` arrête le défaut et `{ECS_AGENT_URI}/fault/v1/network-packet-loss/status` vérifie l'état du défaut. Un seul type de défaut est pris en charge à la fois.

Voici deux exemples de requêtes d'utilisation des points de terminaison `/stop` et `/status`. Les deux utilisent la méthode `POST HTTP`.

```
Endpoint: ${ECS_AGENT_URI}/fault/v1/network-packet-loss/stop
```

```
Endpoint: ${{ECS_AGENT_URI}/fault/v1/network-packet-loss/status
```

Mise à jour des paramètres d'un service Amazon ECS

Après avoir créé un service, il peut arriver que vous deviez mettre à jour les paramètres du service, par exemple le nombre de tâches.

Lorsque le planificateur de service lance de nouvelles tâches, il détermine le placement des tâches dans votre cluster selon la logique suivante.

- Déterminer quelles instances de conteneur de votre cluster peuvent prendre en charge la définition de tâche de votre service. Par exemple, ils disposent des attributs requis en matière de UC, de mémoire, de ports et d'instance de conteneur.
- Par défaut, le planificateur de service tente d'équilibrer les tâches entre les zones de disponibilité de cette manière, même si vous pouvez choisir une autre stratégie de placement.
  - Trier les instances de conteneur valides en fonction du nombre le plus faible de tâches en cours d'exécution pour ce service dans la même zone de disponibilité que l'instance. Par exemple, si la

zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C n'en ont pas, les instances de conteneur valides des zones B ou C sont considérées comme optimales pour le placement.

- Placer la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale (sur la base des étapes précédentes), en préférant les instances de conteneur avec le plus petit nombre de tâches en cours d'exécution pour ce service.

Lorsque le planificateur de service cesse d'exécuter des tâches, il tente de maintenir l'équilibre entre les zones de disponibilité de votre cluster en utilisant la logique suivante :

- Trier les instances de conteneur en fonction du plus grand nombre de tâches en cours d'exécution pour ce service dans la même zone de disponibilité que l'instance. Par exemple, si la zone A comporte une tâche de service en cours d'exécution tandis que les zones B et C en ont chacune deux, les instances de conteneur des zones B ou C sont considérées comme optimales pour la mise hors service.
- Arrêter la nouvelle tâche de service sur une instance de conteneur valide dans une zone de disponibilité optimale (sur la base des étapes précédentes), en préférant les instances de conteneur avec le plus grand nombre de tâches en cours d'exécution pour ce service.

Utilisez la liste pour déterminer si vous pouvez modifier le paramètre de service.

Rééquilibrage des zones de disponibilité

Indique s'il faut utiliser le rééquilibrage des zones de disponibilité pour le service.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Stratégie de fournisseur de capacité

Détails d'une stratégie de fournisseur de capacité. Vous pouvez définir un fournisseur de capacité lorsque vous créez un cluster, exécutez une tâche ou mettez à jour un service.

Lorsque vous utilisez Fargate, les fournisseurs de capacité sont `FARGATE` ou `FARGATE_SPOT`.

Lorsque vous utilisez Amazon EC2, les fournisseurs de capacité sont des groupes Auto Scaling.

Vous pouvez changer de fournisseur de capacité pour les déploiements continus et blue/green déploiements.

Pour faire référence aux transitions valides, voir Mutabilité des services.

Cluster

Vous ne pouvez pas changer le nom du cluster.

Configuration de déploiement

La configuration de déploiement inclut CloudWatch alarmes et disjoncteur utilisés pour détecter les pannes et la configuration requise.

La disjoncteur de circuit de déploiement détermine si un déploiement de service échoue dans le cas où le service ne parvient pas à atteindre un état stable. Si vous utilisez le disjoncteur de déploiement, le déploiement d'un service passera à un état d'échec et cessera de lancer de nouvelles tâches. Si vous utilisez l'option de restauration, lorsqu'un déploiement de service échoue, le service est restauré à la dernière version qui a été déployée avec succès.

Lorsque vous mettez à jour un service qui utilise le disjoncteur Amazon ECS, Amazon ECS crée un déploiement de service et une révision de service. Ces ressources vous permettent de consulter des informations détaillées sur l'historique des services. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Le planificateur de service utilise les paramètres de pourcentage minimum d'instances saines et de pourcentage maximal (dans la configuration de déploiement pour le service) afin de déterminer la stratégie de déploiement.

Si un service utilise le type de déploiement de mise à jour propagée (`ECS`), le pourcentage minimum sain représente la limite inférieure du nombre de tâches d'un service qui doivent rester dans l'état `RUNNING` pendant un déploiement, exprimé en pourcentage du nombre de tâches souhaité (arrondi à l'entier supérieur). Le paramètre s'applique également lorsque des instances de conteneur sont dans l'état `DRAINING` si le service contient des tâches utilisant EC2. Utilisez ce paramètre pour procéder au déploiement sans avoir recours à une capacité de cluster supplémentaire. Par exemple, si votre service comporte un nombre souhaité de tâches égal à quatre et un pourcentage minimum d'instances saines de 50 pour cent, le planificateur pourra arrêter deux tâches existantes afin de libérer de la capacité de cluster avant de lancer deux nouvelles tâches. Le service considère les tâches comme saines pour les services qui n'utilisent pas d'équilibreur de charge si elles sont dans l'état `RUNNING`. Le service considère que les tâches sont saines pour les services qui utilisent un équilibreur de charge si elles sont dans l'état `RUNNING` et si elles sont signalées comme saines par l'équilibreur de charge. La valeur par défaut pour le pourcentage minimum d'instances saines est 100 pour cent.

Si un service utilise le type de déploiement de mise à jour propagée (ECS), le paramètre pourcentage maximal représente une limite supérieure du nombre de tâches dans un service autorisées à avoir l'état PENDING, RUNNING ou STOPPING pendant un déploiement, en tant que pourcentage du nombre de tâches souhaité (arrondi à l'entier le plus proche). Le paramètre s'applique également lorsque des instances de conteneur sont dans l'état DRAINING si le service contient des tâches utilisant EC2. Utilisez ce paramètre pour définir la taille des lots de déploiement. Par exemple, si votre service a un nombre souhaité de quatre tâches et une valeur de pourcentage maximum de 200 pour cent, le planificateur peut lancer quatre nouvelles tâches avant d'arrêter les quatre tâches plus anciennes. Cela s'applique à condition que les ressources de cluster nécessaires à cette opération soient disponibles. La valeur par défaut pour le pourcentage maximal est 200 pour cent.

Lorsque le planificateur de service remplace une tâche pendant une mise à jour, si un équilibreur de charge est utilisé par le service, ce dernier supprime tout d'abord la tâche de l'équilibreur de charge (le cas échéant) et attend que les connexions s'épuisent. Ensuite, l'équivalent de docker stop est émis pour les conteneurs en cours d'exécution dans la tâche. Cela se traduit par un signal SIGTERM et un délai de 30 secondes, après quoi SIGKILL est envoyé et les conteneurs sont arrêtés de force. Si le conteneur gère le signal SIGTERM normalement et s'arrête dans les 30 secondes suivant sa réception, aucun signal SIGKILL n'est envoyé. Le planificateur de service lance et arrête les tâches selon les modalités définies dans vos paramètres de pourcentage minimum d'instances saines et de pourcentage maximal.

Le planificateur de services remplace également les tâches jugées défectueuses après l'échec d'une surveillance de l'état du conteneur ou du groupe cible de l'équilibreur de charge. Ce remplacement dépend des paramètres de définition du service maximumPercent et desiredCount. Si une tâche est marquée comme défectueuse, le planificateur de services lance d'abord une tâche de remplacement. Ensuite, voici ce qui se passe.

- Si l'état de la tâche de remplacement est HEALTHY, le planificateur de service arrête la tâche défectueuse.
- Si l'état de santé de la tâche de remplacement est UNHEALTHY, le planificateur arrête soit la tâche de remplacement défectueuse, soit la tâche défectueuse existante pour que le nombre total de tâches soit égal à desiredCount.

Si le paramètre maximumPercent empêche le planificateur de démarrer d'abord une tâche de remplacement, le planificateur arrête une par une les tâches défectueuses au hasard pour libérer de la capacité, puis lance une tâche de remplacement. Le processus de démarrage et d'arrêt se poursuit jusqu'à ce que toutes les tâches défectueuses soient remplacées par des

tâches saines. Une fois que toutes les tâches défectueuses ont été remplacées et que seules les tâches saines sont en cours d'exécution, si le nombre total de tâches dépasse le `desiredCount`, les tâches saines sont arrêtées au hasard jusqu'à ce que le nombre total de tâches soit égal à `desiredCount`. Pour plus d'informations sur `maximumPercent` et `desiredCount`, veuillez consulter Paramètres de définition de service (langue française non garantie).

Contrôleur de déploiement

Le contrôleur de déploiement à utiliser pour le service. Il existe trois types de contrôleurs de déploiement disponibles :

- `ECS`
- `EXTERNAL`
- `CODE_DEPLOY`

Lorsque vous mettez à jour un service, vous pouvez mettre à jour le contrôleur de déploiement qu'il utilise. La liste suivante fournit les transitions valides :

- Mise à jour depuis CodeDeploy blue/green déploiements (`CODE_DEPLOY`) à ECS Rolling ou blue/green déploiements (`ECS`).
- Mise à jour depuis CodeDeploy blue/green déploiements (`CODE_DEPLOY`) vers des déploiements externes (`EXTERNAL`).
- Mise à jour depuis ECS rolling ou blue/green déploiements (`ECS`) vers des déploiements externes (`EXTERNAL`).
- Mise à jour à partir de déploiements externes (`EXTERNAL`) à ECS Rolling ou blue/green déploiements (`ECS`).

Tenez compte des informations suivantes lorsque vous faites la mise à jour du contrôleur de déploiement d'un service :

- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service depuis le contrôleur de déploiement `ECS` vers un autre contrôleur s'il utilise VPC Lattice ou Amazon ECS Service Connect.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service pendant que le déploiement d'un service est en cours.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service vers `CODE_DEPLOY` s'il n'existe aucun équilibreur de charge sur le service.
- Vous ne pouvez pas mettre à jour le contrôleur de déploiement d'un service depuis `ECS` vers un autre contrôleur si `deploymentConfiguration` inclut des alarmes, un disjoncteur de

déploiement ou une stratégie de déploiement `BLUE_GREEN`. Pour plus d'informations, consultez Contrôleurs et stratégies de déploiement de service Amazon ECS.

- La valeur que vous spécifiez pour `versionConsistency` dans la définition du conteneur ne sera pas utilisée par Amazon ECS si vous mettez à jour le contrôleur de déploiement du service depuis `ECS` vers l'un des autres contrôleurs.
- Si vous mettez à jour le contrôleur de déploiement d'un service depuis `ECS` vers l'un des autres contrôleurs, les réponses de l'API `UpdateService` et `DescribeService` reverront toujours `deployments` au lieu de `taskSets`. Pour plus d'informations sur `UpdateService` et `CreateService`, voir UpdateService et CreateService dans le Référence de l'API Amazon ECS.
- Si un service utilise une stratégie de déploiement de mise à jour propagée, la mise à jour du contrôleur de déploiement depuis `ECS` vers l'un des autres contrôleurs modifiera la façon dont la valeur `maximumPercent` de `deploymentConfiguration` est utilisée. Au lieu d'être simplement utilisé pour limiter le nombre total de tâches dans le cadre d'un déploiement de mise à jour propagée, `maximumPercent` il est utilisé pour remplacer les tâches défectueuses. Pour de plus amples informations sur la manière dont le planificateur remplace les tâches défectueuses, consultez la section Services Amazon ECS.
- Si vous mettez à jour le contrôleur de déploiement d'un service depuis `ECS` vers l'un des autres contrôleurs de déploiement, toute `advancedConfiguration` que vous spécifiez dans la configuration de votre équilibreur de charge sera ignorée. Pour plus d'informations, voir LoadBalancer et AdvancedConfiguration dans le Référence de l'API Amazon ECS.

Lors de la mise à jour du contrôleur de déploiement pour un service utilisant CloudFormation, tenez compte des points suivants en fonction du type de migration que vous effectuez.

- Si vous avez CloudFormation modèle qui contient `EXTERNAL` des informations sur le contrôleur de déploiement ainsi que `TaskSet` et `PrimaryTaskSet` ressources, et vous supprimez les ressources de l'ensemble de tâches du modèle lors de la mise à jour depuis `EXTERNAL` pour `ECS`, le `DescribeTaskSet` et `DeleteTaskSet` Les appels d'API renverront une erreur 400 après la mise à jour du contrôleur de déploiement vers `ECS`. Cela se traduit par une CloudFormation échec de suppression sur les ressources de l'ensemble de tâches, même si CloudFormation empilez les transitions vers `UPDATE_COMPLETE` statut. Pour plus d'informations, consultez la section Ressource retirée de la pile, mais non supprimée dans le Guide de l'utilisateur AWS CloudFormation . Pour résoudre ce problème, supprimez les jeux de tâches directement à l'aide de l'API Amazon ECS `DeleteTaskSet`. Pour plus d'informations sur la suppression d'un ensemble de tâches, voir DeleteTaskSet dans le Amazon Elastic Container Service Référence de l'API.

- Si vous migrez depuis CODE_DEPLOY pour ECS avec une nouvelle définition de tâche et CloudFormation effectue une opération de restauration, Amazon ECS UpdateService la demande échoue avec l'erreur suivante :

```
Resource handler returned message: "Invalid request provided: Unable to update
task definition on services with a CODE_DEPLOY deployment controller. Use AWS
CodeDeploy to trigger a new deployment. (Service: Ecs, Status Code: 400,
Request ID: 0abda1e2-f7b3-4e96-b6e9-c8bc585181ac) (SDK Attempt Count: 1)"
(RequestToken: ba8767eb-c99e-efed-6ec8-25011d9473f0, HandlerErrorCode:
 InvalidRequest)
```

- Après une migration réussie du contrôleur de déploiement ECS vers EXTERNAL, vous devez supprimer manuellement le jeu de tâches ACTIVE, car Amazon ECS ne gère plus le déploiement. Pour plus d'informations sur la suppression d'un ensemble de tâches, voir DeleteTaskSet dans Amazon Elastic Container Service Référence de l'API.

Nombre de tâches souhaité

Le nombre d’instanciations de la tâche à placer et à maintenir en cours d’exécution dans votre service.

Si vous souhaitez arrêter temporairement votre service, définissez cette valeur sur zéro. Ensuite, lorsque vous êtes prêt à démarrer le service, mettez-le à jour avec la valeur d’origine.

Vous pouvez modifier ce paramètre pour les déploiements continus, et blue/green déploiements.

Activer les balises gérées

Indique s’il convient d’activer les balises gérées par Amazon ECS pour les tâches dans le service.

Seules les tâches lancées après la mise à jour refléteront la mise à jour. Pour mettre à jour les balises de toutes les tâches, utilisez l’option de déploiement forcé.

Vous pouvez modifier ce paramètre pour les déploiements continus, et blue/green déploiements.

Activer ECS Exec

Détermine si Amazon ECS Exec est utilisé.

Si vous ne souhaitez pas remplacer la valeur définie lors de la création du service, vous pouvez la définir sur null lorsque vous effectuez cette action.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Période de grâce de la surveillance de l'état

La durée, en secondes, pendant laquelle le planificateur de service Amazon ECS ignore les surveillances de l'état défectueuses d'Elastic Load Balancing, de VPC Lattice et des conteneurs après le premier démarrage d'une tâche. Si vous ne spécifiez pas de valeur pour la période de grâce de la surveillance de l'état, la valeur par défaut `0` est utilisée. Si vous n'utilisez aucune des surveillances de l'état, cela signifie que `healthCheckGracePeriodSeconds` n'est pas utilisé.

Si les tâches de votre service mettent du temps à démarrer et à répondre, vous pouvez spécifier une période de grâce pour la surveillance de l'état pouvant aller jusqu'à 2 147 483 647 secondes (environ 69 ans). Pendant ce temps, le planificateur du service Amazon ECS ignore le statut de la vérification de l'état. La période de grâce peut empêcher le planificateur de services de marquer les tâches comme non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

Vous pouvez modifier ce paramètre pour les déploiements continus et blue/green déploiements.

Équilibreurs de charge

Vous devez utiliser un rôle lié à un service lorsque vous mettez à jour un équilibreur de charge.

Une liste des objets d'équilibreur de charge Elastic Load Balancing. Il contient le nom de l'équilibreur de charge, le nom du conteneur et le port du conteneur pour permettre l'accès à partir de l'équilibreur de charge. Le nom du conteneur est tel qu'il apparaît dans une définition de conteneur.

Amazon ECS ne met pas automatiquement à jour les groupes de sécurité associés aux équilibreurs de charge Elastic Load Balancing ou aux instances de conteneur Amazon ECS.

Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS démarre de nouvelles tâches avec la configuration Elastic Load Balancing mise à jour, puis arrête les anciennes tâches lorsque les nouvelles tâches sont en cours d'exécution.

Pour les services qui utilisent des mises à jour propagées, vous pouvez ajouter, mettre à jour ou supprimer des groupes cibles Elastic Load Balancing. Vous pouvez passer d'un groupe cible unique à plusieurs groupes cibles et inversement.

Pour les services qui utilisent blue/green déploiements, vous pouvez mettre à jour les groupes cibles Elastic Load Balancing en utilisant `CreateDeployment` à travers CodeDeploy. Notez que plusieurs groupes cibles ne sont pas pris en charge pour blue/green déploiements. Pour de plus amples informations, consultez la section Enregistrement de plusieurs groupes cibles auprès d'un service.

Pour les services qui utilisent le contrôleur de déploiement externe, vous pouvez ajouter, mettre à jour ou supprimer des équilibreurs de charge en utilisant CreateTaskSet. Notez que les groupes cibles multiples ne sont pas pris en charge pour les déploiements externes. Pour de plus amples informations, consultez la section Enregistrement de plusieurs groupes cibles auprès d'un service.

Passez une liste vide pour supprimer les équilibreurs de charge.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Configuration réseau

La configuration réseau du service.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Contraintes de placement

Un tableau d'objets de contrainte de placement des tâches permettant de mettre à jour le service à utiliser. Si aucune valeur n'est spécifiée, les contraintes de placement existantes pour le service resteront inchangées. Si cette valeur est spécifiée, elle remplace toutes les contraintes de placement existantes définies pour le service. Pour supprimer toutes les contraintes de placement existantes, spécifiez un tableau vide.

Vous pouvez spécifier un maximum de 10 contraintes par tâche. Cette limite inclut les contraintes de la définition de tâche et celles spécifiées lors de l'exécution.

Vous pouvez modifier ce paramètre pour les déploiements continus, et blue/green déploiements.

Stratégie de placement

Les objets de stratégie de placement permettant de mettre à jour le service à utiliser. Si aucune valeur n'est spécifiée, la stratégie de placement existante pour le service restera inchangée. Si cette valeur est spécifiée, elle remplace la stratégie de placement existante définie pour le service. Pour supprimer une stratégie de placement existante, spécifiez un objet vide.

Vous pouvez modifier ce paramètre pour les déploiements continus, et blue/green déploiements.

Version de la plateforme

La version de plateforme Fargate sur laquelle votre service s'exécute.

Un service utilisant une version de plateforme Linux ne peut pas être mis à jour pour utiliser une version de plateforme Windows et vice versa.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Propager les balises

Indique si la propagation des balises à la tâche doit se faire à partir de la définition de tâche ou à partir du service. Si aucune valeur n'est spécifiée, les balises ne sont pas propagées.

Seules les tâches lancées après la mise à jour refléteront la mise à jour. Pour mettre à jour les balises sur toutes les tâches, définissez `forceNewDeployment` sur `true` afin qu'Amazon ECS démarre de nouvelles tâches avec les balises mises à jour.

Vous pouvez modifier ce paramètre pour les déploiements continus, et blue/green déploiements.

Configuration de Service Connect

La configuration pour Amazon ECS Service Connect. Ce paramètre détermine la manière dont le service se connecte aux autres services de votre application.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Enregistrements de service

Vous devez utiliser un rôle lié à un service lorsque vous mettez à jour les enregistrements de service.

Les détails des enregistrements de découverte de service à attribuer à ce service. Pour de plus amples informations, veuillez consulter Découverte de service.

Lorsque vous ajoutez, mettez à jour ou supprimez la configuration des enregistrements de service, Amazon ECS lance de nouvelles tâches avec la configuration mise à jour des enregistrements de service, puis arrête les anciennes tâches lorsque les nouvelles sont en cours d'exécution.

Transmettez une liste vide pour supprimer les enregistrements de service.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Définition de tâche

La définition et révision de la tâche à utiliser pour le service.

Si vous modifiez les ports utilisés par les conteneurs d'une définition de tâche, vous devrez mettre à jour vos groupes de sécurité d'instances de conteneur afin de fonctionner avec les ports mis à jour.

Si vous mettez à jour la définition de tâche pour le service, le nom et le port de conteneur spécifiés dans la configuration de l'équilibreur de charge doivent rester dans la définition de tâche.

Le comportement d'extraction de l'image du conteneur diffère selon les options de calcul. Pour plus d'informations, consultez :

- Architecte pour AWS Fargate pour Amazon ECS
- Architecture de la capacité EC2 pour Amazon ECS
- Amazon ECS Anywhere) externe pour Amazon ECS

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Configuration du volume

Les détails du volume qui était `configuredAtLaunch`. Lorsque `configuredAtLaunch` est défini sur `true` dans la définition de la tâche, ce paramètre de service configure un volume Amazon EBS pour chaque tâche du service à créer et à associer lors du déploiement. Vous pouvez configurer la taille, le type de volume, les IOPS, le débit, le snapshot et le chiffrement dans ServiceManagedEBSVolumeConfiguration. Le `name` du volume doit correspondre au `name` indiqué dans la définition de la tâche. S'il est défini sur null, aucun nouveau déploiement n'est déclenché. Dans le cas contraire, si cette configuration diffère de celle existante, cela déclenche un nouveau déploiement.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

Configuration de VPC Lattice

La configuration de VPC Lattice pour votre de service. Cela définit comment votre service s'intègre à VPC Lattice pour la communication de service à service.

Vous pouvez modifier ce paramètre pour les déploiements propagés.

AWS CDK Considérations

Le AWS CDK ne suit pas l'état des ressources. Il ne sait pas si vous créez ou mettez à jour un service. Les clients doivent utiliser la trappe d'évacuation pour accéder directement à la structure ecs `Service` L1.

Pour plus d'informations sur les trappes d'évacuation, voir Personnalisez les constructions à partir du AWS Bibliothèque Construct dans le Cloud Development Kit AWS (AWS CDK) Guide du développeur v2.

Pour procéder à la migration de votre service existant vers la structure `ecs.Service`, procédez comme suit :

1. Utilisez la trappe d'évacuation pour accéder à la structure `Service` L1.
2. Définissez manuellement les propriétés suivantes dans la structure `Service` L1.

   Si votre service utilise la capacité Amazon EC2 :

   - `daemon?`
   - `placementConstraints?`
   - `placementStrategies?`
   - Si vous utilisez le mode réseau `awsvpc`, vous devez définir les structures `vpcSubnets?` et `securityGroups?`.

   Si votre service utilise Fargate :

   - `FargatePlatformVersion`
   - Les structures `vpcSubnets?` et `securityGroups?`.
3. Définissez le `launchType` comme suit :

```
const cfnEcsService = service.node.findChild('Service') as ecs.CfnService;
cfnEcsService.launchType = "FARGATE";
```

Pour passer d'un type de lancement à un fournisseur de capacité, procédez comme suit :

1. Utilisez la trappe d'évacuation pour accéder à la structure `Service` L1.
2. Ajoutez la structure `capacityProviderStrategies?`.
3. Déployez le service.

Mettre à jour un service Amazon ECS

Après avoir créé un service, il peut arriver que vous deviez mettre à jour les paramètres du service, par exemple le nombre de tâches.

Lorsque vous mettez à jour un service qui utilise le disjoncteur Amazon ECS, Amazon ECS crée un déploiement de service et une révision de service. Ces ressources vous permettent de consulter des

informations détaillées sur l'historique des services. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Conditions préalables

Avant de mettre à jour un service, vérifiez quels paramètres de service peuvent être modifiés pour votre type de déploiement. Pour obtenir une liste complète de paramètres modifiables, consultez la section Mise à jour des paramètres d'un service Amazon ECS.

Procédure

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, cochez la case à côté du service, puis choisissez Mettre à jour.
4. Pour que votre service démarre un nouveau déploiement, sélectionnez Force new deployment (Forcer un nouveau déploiement).
5. Pour Définition de tâche, choisissez la famille et la révision de définition de tâche.

   > **Important**
   >
   > La console vérifie que la famille et la révision de définition de tâche sélectionnées sont compatibles avec la configuration de calcul définie. Si vous recevez un avertissement, vérifiez la compatibilité de votre définition de tâche et la configuration de calcul que vous avez sélectionnée.

6. Si vous choisissez Replica pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer et à conserver dans le service.
7. Si vous avez choisi Replica, pour qu'Amazon ECS surveille la répartition des tâches entre les zones de disponibilité et les redistribue en cas de déséquilibre, sous Rééquilibrage des services entre zones de disponibilité, sélectionnez Rééquilibrage des services entre zones de disponibilité.
8. Pour Min running tasks (Minimum de tâches en cours d'exécution), saisissez la limite inférieure pour le nombre de tâches du service qui doivent rester à l'état `RUNNING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre

entier supérieur le plus proche). Pour plus d'informations, consultez Deployment configuration (Configuration de déploiement).

9. Pour Max running tasks (Maximum de tâches en cours d'exécution), saisissez la limite supérieure pour le nombre de tâches du service qui peuvent rester à l'état `RUNNING` ou `PENDING` lors d'un déploiement, en tant que pourcentage de nombre souhaité de tâches (arrondi au nombre entier inférieur le plus proche).
10. Pour configurer la manière dont les tâches sont déployées pour votre service, développez Options de déploiement, puis configurez vos options.

    a. Pour Type de contrôleur de déploiement, spécifiez le contrôleur de déploiement de service. La console Amazon ECS prend en charge les types de contrôleurs suivants :`ECS`.

    b. Pour Stratégie de déploiement, choisissez la stratégie utilisée par Amazon ECS pour déployer les nouvelles versions du service.

    c. Selon la Stratégie de déploiement choisie, procédez comme suit :

Stratégie de déploiement	Étapes
Mise à jour continue	a. Pour Pourcentage minimal de tâches en cours d'exécution, spécifiez une valeur minimale en pourcentage du nombre de tâches qui doivent être exécutées pendant le déploiement d'un service. Pour plus d'informations, consultez Déploiement de services Amazon ECS en remplaçant des tâches. b. Pour Pourcentage maximal de tâches

Stratégie de déploiement	Étapes
	en cours d'exécuti on, spécifiez une valeur maximale en pourcentage du nombre de tâches qui peuvent être exécutées pendant le déploieme nt d'un service. Pour plus d'informa tions, consultez Déploiement de services Amazon ECS en remplaçant des tâches.
Blue/green	Pour Temps de cuisson, spécifiez la durée, en minutes, pendant laquelle les révisions de service bleues et vertes doivent être exécutées simultanément. Pour plus d'informations, consultez Amazon ECS blue/green déploiements.

d. Pour exécuter les fonctions Lambda pendant une étape du cycle de vie, sous Crochets liés au cycle de vie procédez comme suit pour chaque fonction Lambda unique :

   i. Choisissez Ajouter.

      Répétez l'opération pour chaque fonction que vous souhaitez exécuter.

   ii. Pour Fonction Lambda, saisissez le nom de la fonction.

   iii. Pour Rôle, choisissez le rôle que vous avez créé dans les prérequis à l'aide du blue/green autorisations.

Pour plus d'informations, consultez Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements.

iv. Pour Étapes du cycle de vie, sélectionnez les étapes exécutées par la fonction Lambda.

v. (Facultatif) Pour Détails du hook, saisissez une paire clé-valeur fournissant des informations sur le hook.

11. En dessous Configuration du déploiement, pour terminer le déploiement avant que la révision du service cible n'atteigne le nombre total souhaité, configurez des critères de réussite précoce. Vous pouvez également choisir le moment où Amazon ECS nettoie les tâches de la révision du service source.

    a. Allumer Critères de réussite précoces.

    b. Pour Pourcentage en bonne santé, entrez le pourcentage de tâches qui doivent être en cours d'exécution et en bon état dans la révision du service cible avant qu'Amazon ECS ne termine le déploiement.

    c. Pour Nettoyage des révisions du service source, choisissez Blocage ou Différé.

12. Pour configurer la manière dont Amazon ECS détecte et gère les échecs de déploiement, développez Deployment failure detection (Détection des échecs de déploiement), puis choisissez vos options.

    a. Pour arrêter un déploiement lorsque les tâches ne peuvent pas démarrer, sélectionnez Use the Amazon ECS deployment circuit breaker (Utiliser le disjoncteur de déploiement Amazon ECS).

    Pour que le logiciel restaure automatiquement le dernier état de déploiement terminé lorsque le disjoncteur de déploiement définit le déploiement comme ayant échoué, sélectionnez Restauration en cas d'échec.

    b. Pour arrêter un déploiement en fonction des statistiques de l'application, sélectionnez Utiliser CloudWatch alarme (s). Puis, à partir de CloudWatch nom de l'alarme, choisissez les alarmes. Pour créer une nouvelle alarme, accédez au CloudWatch console.

    Pour que le logiciel ramène automatiquement le déploiement au dernier état de déploiement terminé lorsqu'un CloudWatch l'alarme met le déploiement en état d'échec, sélectionnez Annulation en cas de panne.

13. Pour modifier les options de calcul, développez Configuration du calcul, puis procédez comme suit :

a. Pour les services sur AWS Fargate, pour Version de la plateforme, choisissez la nouvelle version.

b. Pour les services qui utilisent une stratégie de fournisseur de capacité, pour Stratégie de fournisseur de capacité, procédez comme suit :

- Pour ajouter un fournisseur de capacité supplémentaire, choisissez Ajouter plus. Ensuite, pour Fournisseur de capacité, choisissez le fournisseur de capacité.
- Pour supprimer un fournisseur de capacité, à droite du fournisseur de capacité, choisissez Supprimer.

Un service qui utilise un fournisseur de capacité de groupe Auto Scaling ne peut pas être mis à jour pour utiliser un fournisseur de capacité Fargate. Un service qui utilise un fournisseur de capacité Fargate ne peut pas être mis à jour pour utiliser un fournisseur de capacité de groupe Auto Scaling.

14. (Facultatif) Pour configurer l'autoscaling du service, développez Autoscaling du service, puis spécifiez les paramètres suivants. Pour utiliser l'autoscaling prédictif, qui examine les données de charge passées à partir des flux de trafic, configurez-le après avoir créé le service. Pour plus d'informations, consultez Utilisation des tendances historiques pour mettre à l'échelle les services Amazon ECS via une mise à l'échelle prédictive.

a. Pour utiliser la mise à l'échelle automatique du service, sélectionnez Service Auto Scaling (mise à l'échelle automatique du service).

b. Pour Nombre minimum de tâches, saisissez la limite inférieure du nombre de tâches à utiliser pour l'autoscaling du service. Le nombre souhaité ne sera pas inférieur à ce nombre.

c. Pour Nombre maximal de tâches, saisissez la limite supérieure du nombre de tâches à utiliser par l'autoscaling du service. Le nombre souhaité ne sera pas supérieur à ce nombre.

d. Choisissez le type de stratégie. Sous Type de stratégie de mise à l'échelle, choisissez l'une des options suivantes.

Pour utiliser ce type de stratégie	Faites ceci	
Suivi des cibles	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Target tracking (Suivi de cible). b. Pour Policy name (Nom de la politique), saisissez un nom de politique. c. Pour Métrique de service ECS, sélectionnez l'une des métriques suivantes. • ECSServiceAverageCPUUtilization — Utilisation moyenne du processeur du service. • ECSServiceAverageMemoryUtilization — Utilisation moyenne de la mémoire du service. • ALBRequestCountPerTarget — Nombre de demandes traitées par cible dans un groupe cible Application Load Balancer.	

Pour utiliser ce type de stratégie	Faites ceci	
	d. PourTarget value (Valeur cible), entrez la valeur conservée par le service pour la métrique sélectionnée. e. Pour Scale-out période de refroidissement, entrez le temps, en secondes, qui s'écoule après une activité de scale-out (ajout de tâches) qui doit s'écouler avant qu'une autre activité de scale-out puisse démarrer. f. Pour Scale-in période de refroidissement, entrez le temps, en secondes, qui s'écoule après une activité d'extension (suppress ion de tâches) avant qu'une autre activité d'extension puisse démarrer. g. Pour empêcher la politique d'effectuer une activité de mise à l'échelle horizonta le, sélectionnez Turn off scale-in (Désactiv er la mise à l'échelle horizontale).	

Pour utiliser ce type de stratégie	Faites ceci	
	h. • (Facultatif) Sélectionnez Désactiver la mise à l'échelle horizontale si vous souhaitez que votre stratégie de mise à l'échelle augmente horizontalement en cas de hausse du trafic, mais ne doive pas réduire horizontalement lorsque le trafic diminue	

Pour utiliser ce type de stratégie	Faites ceci
Mise à l'échelle par étapes	a. Pour Scaling policy type (Type de politique de mise à l'échelle), choisissez Mise à l'échelle par étapes. b. Pour Nom de la stratégie, saisissez un nom de stratégie. c. Pour Alarm name (Nom de l'alarme), entrez un nom unique pour l'alarme. d. Pour Métrique de service Amazon ECS, sélectionnez la métrique à utiliser pour l'alarme. e. Pour Statistique, choisissez la statistique d'alarme. f. Pour Période, choisissez la période de l'alarme. g. Pour Condition d'alarme, choisissez comment comparer la métrique sélectionnée au seuil défini. h. Pour Seuil de comparaison des métriques et Période d'évaluation pour déclencher l'alarme,

Pour utiliser ce type de stratégie	Faites ceci	
	saisissez le seuil utilisé pour l'alarme et la durée d'évaluation du seuil. i. Sous Actions de mise à l'échelle, procédez comme suit : • Pour Action, sélectionnez si vous souhaitez ajouter, supprimer ou définir un nombre spécifique souhaité pour votre service. • Si vous avez choisi d’ajouter ou de supprimer des tâches, pour Valeur, saisissez le nombre de tâches (ou le pourcentage de tâches existantes) à ajouter ou à soustraire lors du lancement de l’action de mise à l’échelle. Si vous avez choisi de spécifier le nombre souhaité, saisissez le nombre de tâches. Pour Type, indiquez si la Valeur est un entier ou un pourcentage	

Pour utiliser ce type de stratégie	Faites ceci	
	du nombre souhaité existant. • Pour Limite inférieure et Limite supérieure, saisissez les limites inférieure et supérieure de votre ajustement de mise à l'échelle par étapes. Par défaut, la limite inférieure pour l'ajout d'une politique est le seuil de l'alarme et la limite supérieure est l'infini positif (+). Par défaut, la limite supérieure pour la suppression d'une politique est le seuil de l'alarme et la limite inférieure est l'infini négatif (-). • (Facultatif) Ajoutez des options de mise à l'échelle supplémentaires. Choisissez Ajouter une nouvelle action de mise à l'échelle, puis répétez les étapes de l'Action de mise à l'échelle. • Pour Temps de stabilisation,	

Pour utiliser ce type de stratégie	Faites ceci	
	saisissez la durée, en secondes, à attendre avant qu'une activité de mise à l'échelle précédent e ne prenne effet. Pour une stratégie d'ajout, il s'agit du délai après une activité d'augment ation horizontale pendant lequel la stratégie de mise à l'échelle bloque les activités de réduction horizontale et limite le nombre de tâches pouvant augmenter horizontalement simultanément. Pour une politique de suppression, il s'agit du délai qui doit s'écouler après une activité de réduction horizontale avant qu'une autre activité de réduction puisse commencer.	

15. (Facultatif) Pour utiliser Service Connect, sélectionnez Turn on Service Connect (Activer Service Connect), puis spécifiez les informations suivantes :

    a. Sous Service Connect configuration (Configuration de Service Connect), spécifiez le mode client.

- Si votre service exécute une application client réseau qui doit uniquement se connecter à d'autres services de l'espace de noms, sélectionnez Côté client uniquement.
- Si votre service exécute une application réseau ou un service Web et doit fournir des points de terminaison pour ce service, et se connecte à d'autres services dans l'espace de noms, choisissez Client and server (Client et serveur).

b. Pour utiliser un espace de noms qui n'est pas l'espace de noms de cluster par défaut, dans Namespace (Espace de noms), choisissez l'espace de noms du service. Il peut s'agir d'un espace de noms créé séparément dans le même Région AWS dans votre Compte AWS ou un espace de noms dans la même région qui est partagé avec votre compte à l'aide de AWS Resource Access Manager (AWS RAM). Pour plus d'informations sur le partage AWS Cloud Map espaces de noms, voir Cross-account AWS Cloud Map partage d'espaces de noms dans le AWS Cloud Map Guide du développeur

c. (Facultatif) Spécifiez une configuration de journal. Sélectionnez Utiliser la collecte de journaux. L'option par défaut envoie les journaux des conteneurs à CloudWatch Journaux. Les autres options du pilote de journalisation sont configurées à l'aide de AWS FireLens. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.

   Voici une description plus détaillée de chaque destination de journal de conteneur.

   - Amazon CloudWatch — Configurez la tâche à laquelle envoyer les journaux des conteneurs CloudWatch Journaux. Les options du pilote de journal par défaut sont fournies, ce qui crée un CloudWatch connectez-vous à un groupe en votre nom. Pour spécifier un autre nom de groupe de journaux, modifiez les valeurs des options de pilote.
   - Amazon Data Firehose : configurez la tâche pour envoyer des journaux de conteneur à Firehose. Les options par défaut du pilote de journalisation sont fournies. Elles envoient les journaux vers un flux de livraison Firehose. Pour spécifier un autre nom de flux de diffusion, modifiez les valeurs des options de pilote.
   - Amazon Kinesis Data Streams : configurez la tâche pour envoyer des journaux de conteneur à Kinesis Data Streams. Les options par défaut du pilote de journalisation sont fournies. Elles permettent d'envoyer les journaux vers un flux Kinesis Data

Streams. Pour spécifier un autre nom de flux, modifiez les valeurs des options de pilote.

- Amazon OpenSearch Service — Configurez la tâche pour envoyer les journaux des conteneurs à un OpenSearch Domaine de service. Les options de pilote de journal doivent être fournies.
- Simple Storage Service (Amazon S3) : configurez la tâche pour envoyer des journaux de conteneur à un compartiment Simple Storage Service (Amazon S3). Les options du pilote de journal par défaut sont définies par défaut, mais vous devez spécifier un nom de compartiment Amazon S3 valide.

d. Pour activer les journaux d'accès, procédez comme suit :

i. Élargir Configuration du journal d'accès. Pour Formater, choisissez soit JSON ou TEXT.

ii. Pour inclure les paramètres de requête dans les journaux d'accès, sélectionnez Inclure les paramètres de requête.

> **Note**
>
> Pour désactiver les journaux d'accès, pour Formater, choisissez Aucune.

16. Si votre tâche utilise un volume de données compatible avec la configuration lors du déploiement, vous pouvez configurer le volume en développant Volume.

    Le nom et le type de volume sont configurés lorsque vous créez une révision de définition de tâche et ne peuvent pas être modifiés lorsque vous mettez à jour un service. Pour mettre à jour le nom et le type du volume, vous devez créer une révision de définition de tâche et mettre à jour un service à l'aide de la nouvelle révision.

Pour configurer ce type de volume	Faites ceci	
Amazon EBS	a. Pour Type de volume EBS, choisissez le type de volume EBS que vous souhaitez attacher à votre tâche.	

Pour configurer ce type de volume	Faites ceci	
	b. Pour Taille (Gio), saisissez une valeur valide pour la taille du volume en gibioctet s (Gio). Vous pouvez spécifier une taille de volume minimale de 1 Gio et maximale de 16 384 Gio. Cette valeur est obligatoire sauf si vous fournissez un ID d'instantané. c. Pour IOPS, entrez le nombre maximum de input/output opération s (IOPS) que le volume doit fournir. Cette valeur n'est configurable que pour les types de volume `io1`, `io2` et `gp3`. d. Pour Débit (MiB/s), entrez le débit que le volume doit fournir, en mébioctets par seconde (MiBps, ou MiB/s). Cette valeur n'est configura ble que pour le type de volume `gp3`. e. Pour ID d'instantané, choisissez un instantané de volume Amazon EBS existant ou saisissez l'ARN d'un instantan	

Pour configurer ce type de volume	Faites ceci	
	é si vous souhaitez créer un volume à partir d'un instantané. Vous pouvez également créer un volume vide en ne choisissant ni ne saisissant aucun ID d'instantané. f. Si vous spécifiez un Identifiant de l'instantané, vous pouvez spécifier Taux d'initialisation du volume (MiB/s). Entrez une valeur comprise entre 100 et 300, dans MiB/s, qui déterminera la vitesse de chargemen t des données à partir de l'instantané spécifié à l'aide de Identifiant de l'instantané pour la création de volumes. g. Pour Type de système de fichiers, choisisse z le type de système de fichiers qui sera utilisé pour le stockage et la récupération des données sur le volume. Vous pouvez choisir le système d'exploit ation par défaut ou un type de système de	

Pour configurer ce type de volume	Faites ceci	
	fichiers spécifique. La valeur par défaut pour Linux est `XFS`. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarre pas. h. Pour Rôle d'infrastructure, choisissez un rôle IAM doté des autorisations nécessaires pour permettre à Amazon ECS de gérer les volumes Amazon EBS pour les tâches. Vous pouvez associer la politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` au rôle, ou vous pouvez utiliser la politique comme guide pour créer et associer votre propre politique avec des autorisations répondant	

Pour configurer ce type de volume	Faites ceci	
	à vos besoins spécifiques. Pour plus d'informations sur les autorisations nécessaires, consultez Rôle IAM d'infrastructure Amazon ECS. i. Pour Chiffrement, sélectionnez Par défaut si vous souhaitez utiliser le chiffrement Amazon EBS par défaut. Si votre compte possède Chiffrement par défaut configuré, le volume sera chiffré avec AWS Key Management Service (AWS KMS) touche spécifiée dans le paramètre. Si vous choisissez Par défaut et que le chiffrement par défaut d'Amazon EBS n'est pas activé, le volume ne sera pas chiffré. Si tu choisis Personnalisé, vous pouvez spécifier un AWS KMS key de votre choix pour le chiffrement des volumes.	

Pour configurer ce type de volume	Faites ceci	
	Si vous choisissez Aucun, le volume ne sera pas chiffré, sauf si le chiffrement par défaut est configuré ou si vous créez un volume à partir d'un instantané chiffré. j. Si vous avez choisi Personnalisé pour Chiffrement, vous devez spécifier AWS KMS key que vous souhaitez utiliser. Pour la clé KMS, choisissez une AWS KMS key ou saisissez un ARN. Si vous choisissez de chiffrer votre volume à l'aide d'une clé symétrique gérée par le client, assurez-vous que vous disposez des autorisations appropriées définies dans votre AWS KMS key politique. Pour plus d'informations, consultez la section Chiffrement des données pour les volumes Amazon EBS. k. (Facultatif) Sous Balises, vous pouvez ajouter des balises à votre volume Amazon EBS soit en propageant des balises à	

Pour configurer ce type de volume	Faites ceci	
	partir de la définition de la tâche ou du service, soit en fournissant vos propres balises. Si vous souhaitez propager des balises à partir de la définitio n de tâche, choisisse z Définition de tâche pour Propager des balises à partir de. Si vous souhaitez propager des balises depuis le service, choisissez Service pour Propager des balises à partir de. Si vous choisissez Ne pas propager ou si vous ne choisissez aucune valeur, les balises ne sont pas propagées. Si vous souhaitez fournir vos propres balises, choisissez Ajouter une balise, puis indiquez la clé et la valeur pour chaque balise que vous ajoutez. Pour plus d’informa tions sur la création d’es volumes Amazon EBS, consultez la section	

Pour configurer ce type de volume	Faites ceci	
	Étiquetage d'un volume Amazon EBS.	

17. (Facultatif) Pour vous aider à identifier votre service, développez Tags (Balises), puis configurez vos balises.
    - [Ajouter une balise] Choisissez Ajouter une balise et procédez comme suit :
      - Pour Clé, saisissez le nom de la clé.
      - Pour Valeur, saisissez la valeur de clé.
    - [Supprimer une balise] En regard de la balise, choisissez Supprimer la balise.
18. Choisissez Mettre à jour.

AWS CLI

- Exécutez `update-service`. Pour plus d'informations sur l'exécution de la commande, voir service de mise à jour dans le AWS Command Line Interface Référence.

  L'exemple `update-service` suivant met à jour le nombre de tâches souhaité du service `my-http-service` sur deux.

  Remplacez le *`user-input`* avec tes valeurs.

```
aws ecs update-service \
    --cluster MyCluster \
    --service my-http-service \
    --desired-count 2
```

Étapes suivantes

Suivez votre déploiement et consultez l'historique des services pour les services qui utilisent le disjoncteur Amazon ECS. Pour plus d'informations, consultez Affichage de l'historique d'un service à l'aide des déploiements de service Amazon ECS.

Mise à jour d'un service Amazon ECS pour utiliser un fournisseur de capacité

Si vous avez un service existant qui utilise le type de lancement Amazon EC2 ou Fargate et que vous souhaitez utiliser les instances gérées Amazon ECS, vous devez mettre à jour le service pour utiliser votre fournisseur de capacité d'instances gérées Amazon ECS.

Conditions préalables

Créez un fournisseur de capacité pour vos instances gérées Amazon ECS. Pour plus d'informations, consultez Création d'un fournisseur de capacité pour les instances gérées Amazon ECS.

Procédure

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, cochez la case à côté du service, puis choisissez Mettre à jour.
4. Sélectionnez Forcer un nouveau déploiement.
5. Sous Configuration de calcul, choisissez la stratégie de fournisseur de capacité. Choisissez ensuite l'une des méthodes suivantes :
   - Lorsque votre fournisseur de capacité d'instances gérées Amazon ECS est le fournisseur de capacité par défaut, choisissez Utiliser le cluster par défaut.
   - Lorsque votre fournisseur de capacité d'instances gérées Amazon ECS n'est pas le fournisseur de capacité par défaut, choisissez Utiliser un cluster personnalisé (Avancé). Choisissez votre fournisseur de capacité d'instances gérées Amazon ECS, puis pour Poids, choisissez 1.
6. Choisissez Mettre à jour.

AWS CLI

- Exécutez `update-service`. Pour plus d'informations sur l'exécution de la commande, voir service de mise à jour dans le AWS Command Line Interface Référence.

  Remplacez le *`user-input`* avec tes valeurs.

```
aws ecs update-service \
    --cluster my-cluster \
    --service my-service \
    --capacity-provider-strategy capacityProvider=my-managed-instance-capacity-provider,weight=1 \
    --force-new-deployment
```

Suppression d’un service Amazon ECS à l’aide de la console

Voici quelques-unes des raisons pour lesquelles vous voudriez supprimer un service :

- L’application n’est plus nécessaire
- Vous procédez à la migration du service vers un nouvel environnement
- L’application n’est pas utilisée activement
- L’application utilise plus de ressources que nécessaire et vous essayez d’optimiser vos coûts

La capacité du service est automatiquement réduite à 0. Les ressources de l'équilibreur de charge ou de la découverte de service qui sont associées au service, elles ne sont pas affectées par la suppression de service. Pour supprimer vos ressources Elastic Load Balancing, consultez l'une des rubriques suivantes, en fonction du type de votre équilibreur de charge : Suppression d'un Application Load Balancer ou Suppression d'un Network Load Balancer.

Lorsque vous supprimez un service, Amazon ECS supprime tous les déploiements de service et toutes les révisions de service pour ce service.

Lorsque vous supprimez un service, s’il reste des tâches en cours d’exécution qui nécessitent un nettoyage, le statut du service passe de `ACTIVE` à `DRAINING`, et le service n’est plus visible dans la console ni dans l’opération d’API `ListServices`. Une fois que toutes les tâches sont passées à l'état `STOPPING` ou `STOPPED`, l'état du service passe de `DRAINING` à `INACTIVE`. Les services ayant le statut `DRAINING` ou `INACTIVE` peuvent toujours être consultés à l’aide de l’opération d’API `DescribeServices`.

> **Important**
>
> Si vous essayez de créer un nouveau service portant le même nom qu’un service existant dont le statut est `ACTIVE` ou `DRAINING`, vous recevrez un message d’erreur.

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, sélectionnez le cluster pour le service.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Des services onglet.
5. Sélectionnez les services, puis choisissez Delete (Supprimer).
6. Pour supprimer un service même s'il n'a pas été réduit à zéro tâche, sélectionnezForce delete service (Forcer la suppression du service).
7. À l'invite de confirmation, saisissez supprimer, puis choisissez Supprimer.

Migration d'un ARN court de service Amazon ECS vers un ARN long

Amazon ECS attribue un Amazon Resource Name (ARN) unique à chaque service. Les services créés avant 2021 ont un format ARN court :

`arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`service-name`*

Amazon ECS a modifié le format ARN pour inclure le nom du cluster. Ceci est un format ARN long :

`arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

Pour pouvoir baliser votre service, celui-ci doit avoir un format d'ARN long.

Vous pouvez procéder à la migration d'un service au format ARN court vers le format ARN long sans avoir à recréer le service. Vous pouvez utiliser l'API, la CLI ou la console. Vous ne pouvez pas annuler l'opération de migration.

Le processus de migration est transparent et garantit une durée d'indisponibilité nulle pour votre service. Pendant la migration :

- Disponibilité du service : votre service continue de fonctionner normalement sans interruption du trafic ou des fonctionnalités.
- Tâches en cours : les tâches existantes continuent de s'exécuter sans interruption. Les nouvelles tâches lancées après la migration utiliseront le format ARN long si le paramètre du compte `taskLongArnFormat` est activé.
- Instances de conteneur : les instances de conteneur ne sont pas affectées par la migration de l'ARN du service et continuent de fonctionner normalement.

- Configuration du service : tous les paramètres du service, y compris la définition des tâches, la mise en réseau et les configurations de l'équilibreur de charge, restent inchangés.

Si vous souhaitez utiliser CloudFormation pour baliser un service au format ARN court, vous devez migrer le service à l'aide de l'API, de l'interface de ligne de commande ou de la console. Une fois la migration terminée, vous pouvez utiliser CloudFormation pour étiqueter le service.

Si vous souhaitez utiliser Terraform pour étiqueter un service avec un format ARN court, vous devez procéder à la migration du service à l'aide de l'API, de la CLI ou de la console. Une fois la migration terminée, vous pouvez utiliser Terraform pour étiqueter le service.

Une fois la migration terminée, le service comporte les changements suivants :

- Le format ARN long

  `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`cluster-name`*`/`*`service-name`*

- Lorsque vous effectuez une migration à l'aide de la console, Amazon ECS ajoute une balise au service dont la clé est définie sur « ecs:service ».ArnMigratedAt« et la valeur définie sur l'horodatage de la migration (format UTC).

  Cette balise est prise en compte dans votre quota de balises.

- Quand le `PhysicalResourceId` dans un CloudFormation stack représente un ARN de service, la valeur ne change pas et continuera d'être l'ARN de service court.

Conditions préalables

Effectuez les opérations suivantes avant de procéder à la migration de l'ARN du service.

1. Pour savoir si vous avez un ARN court de service, consultez les détails du service dans la console Amazon ECS (un avertissement s'affiche lorsque le service utilise le format d'ARN court) ou le paramètre de retour `serviceARN` renvoyé par `describe-services`. Lorsque l'ARN n'inclut pas le nom du cluster, il s'agit d'un ARN court. Voici le format d'un ARN court :

   `arn:aws:ecs:`*`region`*`:`*`aws_account_id`*`:service/`*`service-name`*

2. Notez la date de création.
3. Si vous avez des politiques IAM qui utilisent le format ARN court, mettez-les à jour pour passer au format ARN long.

Remplacez chaque *user input placeholder* par vos propres informations.

```
arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
```

Pour plus d'informations, voir Modifier les politiques IAM dans le Gestion des identités et des accès AWS Guide de l'utilisateur.

4. Si vous avez des outils qui utilisent le format ARN court, mettez-les à jour pour passer au format ARN long.

   Remplacez chaque *user input placeholder* par vos propres informations.

   ```
   arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
   ```

5. Activer le format ARN long du service. Exécutez `put-account-setting` avec l'option `serviceLongArnFormat` définie à `enabled`. Pour plus d'informations, consultez la section put-account-setting dans la Référence de l'API Amazon Elastic Container Service.

   Exécutez la commande en tant qu'utilisateur racine lorsque votre service a une date `createdAt` inconnue.

   ```
   aws ecs put-account-setting --name serviceLongArnFormat --value enabled
   ```

   Exemple de sortie

   ```
   {
       "setting": {
           "name": "serviceLongArnFormat",
           "value": "enabled",
           "principalArn": "arn:aws:iam::123456789012:role/your-role",
           "type": user
       }
   }
   ```

6. Activer le format ARN long de la tâche. Ce paramètre de compte contrôle le format ARN pour les nouvelles tâches lancées une fois la migration du service terminée. Exécutez `put-account-setting` avec l'option `taskLongArnFormat` définie à `enabled`. Pour plus d'informations, consultez la section put-account-setting dans la Référence de l'API Amazon Elastic Container Service.

Exécutez la commande en tant qu'utilisateur racine lorsque votre service a une date `createdAt` inconnue.

```
aws ecs put-account-setting --name taskLongArnFormat --value enabled
```

Exemple de sortie

```
{
    "setting": {
        "name": "taskLongArnFormat",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:role/your-role",
        "type": user
    }
}
```

> **Note**
>
> Le paramètre `taskLongArnFormat` ne déclenche pas directement la migration des tâches existantes. Il n'affecte que le format ARN des nouvelles tâches créées après l'activation du paramètre. Les tâches en cours d'exécution existantes conservent leur format ARN actuel jusqu'à ce qu'elles soient remplacées dans le cadre d'opérations de service normales (telles que des déploiements ou des activités de mise à l'échelle).

Procédure

Suivez les instructions suivantes pour procéder à la migration de votre ARN de service.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Dans la section Services, choisissez un service dont la colonne ARN contient un avertissement.

   La page de détails du service s'affiche.
4. Choisissez Procéder à la migration vers un ARN long.

La boîte de dialogue de « Procéder à la migration du service » s'affiche.

5. Choisissez Migrate (Migrer).

INTERFACE DE LIGNE DE COMMANDE (CLI)

Une fois les conditions préalables remplies, vous pouvez étiqueter votre service. Exécutez la commande suivante :

Amazon ECS envisage de transmettre le format d'ARN long dans une requête d'API `tag-resource` pour un service avec un ARN court comme signal de migration du service afin d'utiliser le format d'ARN long.

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:region:aws_account_id:service/cluster-name/service-name
    --tags key=key1,value=value1
```

Les exemples de balises suivants MyService avec une étiquette dont la clé est réglée sur »TestService« et une valeur définie sur »WebServers:

```
aws ecs tag-resource \
    --resource-arn arn:aws:ecs:us-east-1:123456789012:service/MyCluster/MyService
    --tags key=TestService1,value=WebServers
```

Terraform

Une fois les conditions préalables remplies, vous pouvez étiqueter votre service. Créez une ressource `aws_ecs_service` et définissez la référence `tags`. Pour plus d'informations, consultez la section Resource : aws_ecs_service dans la documentation de Terraform.

```
resource "aws_ecs_service" "MyService" {
  name    = "example"
  cluster = aws_ecs_cluster.MyService.id

 tags = {
 "Name"  =  "MyService"
 "Environment"  =  "Production"
 "Department"  =  "QualityAssurance"
  }
```

```
}
```

Étapes suivantes

Vous pouvez ajouter des balises au service. Pour plus d'informations, consultez Ajout de balises aux ressources Amazon ECS.

Si vous souhaitez qu'Amazon ECS propage les balises de la définition de tâche ou du service dans la tâche, exécutez `update-service` avec le paramètre `propagateTags`. Pour plus d'informations, voir service de mise à jour dans le AWS Command Line Interface Référence.

Résolution des problèmes

Certains utilisateurs peuvent rencontrer l'erreur suivante lorsqu'ils procèdent à la migration du format d'ARN court vers le format d'ARN long.

`There was an error while migrating the ARN of service` *`service-name`*`. The specified account does not have serviceLongArnFormat or taskLongArnFormat account settings enabled. Add account settings in order to enable tagging.`

Si vous avez déjà activé les paramètres du compte `serviceLongArnFormat`, mais que vous rencontrez toujours cette erreur, cela peut être dû au fait que les paramètres du compte pour le format ARN long n'ont pas été activés pour le principal IAM qui a initialement créé le service.

1. Identifiez le principal qui a créé le service.

    1. Dans la console, les informations sont disponibles dans le champ Créé par de l'onglet Configuration et mise en réseau de la page « Détails du service » de la console Amazon ECS.
    2. Pour le AWS CLI, exécutez la commande suivante :

        Remplacez le *`user-input`* avec tes valeurs.

        ```
        aws ecs describe-services --cluster cluster-name --services service-name --query
         'services[0].{createdBy: createdBy}'
        ```

2. Activez les paramètres de compte requis pour ce principal spécifique. Vous pouvez effectuer cette opération de différentes manières :

    a. Assumez le rôle ou l'utilisateur IAM pour ce principal. Ensuite, exécutez `put-account-setting`.

b. Utilisez l'utilisateur racine pour exécuter la commande tout en spécifiant le principal créateur avec le `principal-arn`.

   Exemple.

   Remplacez le *`principal-arn`* avec la valeur de l'étape 1.

```
aws ecs put-account-setting --name serviceLongArnFormat --value enabled --
principal-arn arn:aws:iam::123456789012:role/jdoe
```

Les deux méthodes activent le paramètre de compte `serviceLongArnFormat` requis sur le principal qui a créé le service, ce qui permet de procéder à la migration de l'ARN.

Logique de limitation d'un service Amazon ECS

Le planificateur de service Amazon ECS inclut désormais une logique qui limite la fréquence de lancement des tâches de service en cas d'échec de lancement répété. Cela permet d'éviter une consommation inutile de ressources et de réduire les coûts.

Lorsque les tâches d'un service ne passent pas d'un état `PENDING` à `RUNNING` et passent directement à un état `STOPPED`, le planificateur :

- augmente progressivement le délai entre les tentatives de redémarrage ;
- continue d'augmenter les délais jusqu'à 27 minutes entre les tentatives ;
- génère un message d'événement de service pour vous informer du problème.

> **Note**
>
> Le délai maximum de 27 minutes peut changer dans les futures mises à jour.

Lorsque la limitation est activée, vous recevez ce message d'événement de service :

```
(service service-name) is unable to consistently start tasks successfully.
```

Caractéristiques importantes de la logique de limitation :

- Les services poursuivent indéfiniment les tentatives

- La seule modification est l'augmentation du temps entre les redémarrages
- Il n'existe aucun paramètre configurable par l'utilisateur

Résolution des problèmes de limitation

Pour résoudre le problème de limitation, vous pouvez :

- mettre à jour le service afin qu'il utilise une nouvelle définition de tâche, ce qui rétablit immédiatement le fonctionnement normal et non limité du service ; Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.
- corriger la cause sous-jacente des défaillances de tâches.

Les causes courantes de défaillances de tâches qui déclenchent la limitation sont les suivantes :

- Ressources de cluster insuffisantes (ports, mémoire ou UC)
  - Signalé par un message d'événement de service de ressources insuffisantes
- Échecs d'extraction de l'image de conteneur
  - Cela peut être dû à des noms d'image non valides, à des balises ou à des autorisations insuffisantes
  - Résulte en `CannotPullContainerError` dans Affichage des erreurs liées aux tâches Amazon ECS arrêtées
- Espace disque insuffisant
  - Résulte en `CannotCreateContainerError` dans les erreurs liées aux tâches arrêtées
  - Pour les étapes de résolution, consultez la section Résoudre les problèmes liés au Docker `Erreur d'API (500) : devmapper` dans Amazon ECS

> **Important**
>
> Les scénarios suivants ne déclenchent PAS la logique de limitation :
>
> - Tâches qui s'arrêtent après avoir atteint l'état `RUNNING`
> - Tâches arrêtées suite à l'échec des surveillances d'état d'Elastic Load Balancing
> - Tâches pour lesquelles la commande du conteneur se termine avec un code différent de zéro après avoir atteint l'état `RUNNING`

Paramètres de définition d’un service Amazon ECS

Une définition de service définit comment exécuter votre service Amazon ECS service. Les paramètres suivants peuvent être spécifiés dans une définition de service.

Type de lancement

`launchType`

Type : String

Valeurs valides : `EC2` | `FARGATE` | `EXTERNAL`

Obligatoire : non

Type de lancement sur lequel vous pouvez exécuter votre service. Si aucun type de lancement n'est spécifié, the `capacityProviderStrategy` par défaut est utilisé par défaut.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

Si un `launchType` est spécifié, le paramètre `capacityProviderStrategy` doit être omis.

Stratégie de fournisseur de capacité

`capacityProviderStrategy`

Type : tableau d'objets

Obligatoire : non

Stratégie du fournisseur de capacité à utiliser pour le service.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d’un nouveau service.

Une stratégie de fournisseur de capacité consiste en un ou plusieurs fournisseurs de capacité avec le `base` et `weight` à leur attribuer. Un fournisseur de capacité doit être associé au cluster à utiliser dans une stratégie de fournisseur de capacité. Le PutClusterCapacityProviders L'API est utilisée pour associer un fournisseur de capacité à un cluster. Seuls les fournisseurs de capacité ayant un statut `ACTIVE` ou `UPDATING` peuvent être utilisés.

Si un `capacityProviderStrategy` est spécifié, le paramètre `launchType` doit être omis. Lorsque ni `capacityProviderStrategy` ni `launchType` ne sont spécifiés, le `defaultCapacityProviderStrategy` pour le cluster est utilisé.

Si vous souhaitez spécifier un fournisseur de capacité qui utilise un groupe Auto Scaling, le fournisseur de capacité doit déjà être créé. De nouveaux fournisseurs de capacités peuvent être créés grâce au CreateCapacityProvider Fonctionnement de l'API.

Pour utiliser un AWS Fournisseur de capacité Fargate, spécifiez soit le `FARGATE` ou `FARGATE_SPOT` fournisseurs de capacités. Le AWS Les fournisseurs de capacité Fargate sont disponibles pour tous les comptes et doivent uniquement être associés à un cluster pour être utilisés.

Le PutClusterCapacityProviders L'opération d'API est utilisée pour mettre à jour la liste des fournisseurs de capacité disponibles pour un cluster après sa création.

`capacityProvider`

Type : chaîne

Obligatoire : oui

Le nom abrégé ou l'Amazon Resource Name (ARN) complet du fournisseur de capacité.

`weight`

Type : Integer

Plage valide : entiers compris entre 0 et 1 000.

Obligatoire : non

La valeur de poids indique le pourcentage relatif du nombre total de tâches lancées qui utilisent le fournisseur de capacité spécifié.

Par exemple, supposons que vous avez une stratégie qui contient deux fournisseurs de capacité et que les deux ont un poids de un. Lorsque la base est satisfaite, les tâches sont réparties équitablement entre les deux fournisseurs de capacité. Dans la même logique, supposons que vous spécifiez un poids de 1 pour capacityProviderA et un poids de 4 pour capacityProviderB. Ensuite, pour chaque tâche exécutée avec capacityProviderA, quatre tâches utilisent capacityProviderB.

`base`

Type : Integer

Plage valide : entiers compris entre 0 et 100 000.

Obligatoire : non

La valeur de base indique le nombre minimum de tâches à exécuter sur le fournisseur de capacité spécifié. Une base ne peut être définie que pour un seul fournisseur de capacité dans une stratégie de fournisseur de capacité.

Définition de tâche

`taskDefinition`

Type : chaîne

Obligatoire : non

L'Amazon Resource Name (ARN) complet ou `family` et `revision` (`family:revision`) de la définition de tâche à exécuter dans votre service. Si un `revision` n'est pas spécifié, la dernière révision `ACTIVE` de la famille spécifiée est utilisée.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

Une définition de tâche doit être spécifiée lors de l'utilisation du contrôleur de déploiement de mise à jour continue (`ECS`).

Système d'exploitation de la plateforme

`platformFamily`

Type : chaîne

Obligatoire : Conditionnelle

Par défaut : Linux

Ce paramètre est requis pour les services Amazon ECS services hébergés sur Fargate.

Ce paramètre est ignoré pour les services Amazon ECS services hébergés sur Amazon EC2.

Système d'exploitation des conteneurs qui exécutent le service. Les valeurs valides sont `LINUX`, `WINDOWS_SERVER_2019_FULL`, `WINDOWS_SERVER_2019_CORE`, `WINDOWS_SERVER_2022_FULL` et `WINDOWS_SERVER_2022_CORE`.

La valeur `platformFamily` pour chaque tâche que vous spécifiez pour le service doit correspondre à la valeur `platformFamily` du service. Par exemple, si vous définissez `platformFamily` sur `WINDOWS_SERVER_2019_FULL`, la valeur `platformFamily` de toutes les tâches doit être `WINDOWS_SERVER_2019_FULL`.

Version de plateforme

`platformVersion`

Type : chaîne

Obligatoire : non

Version de la plateforme sur laquelle s'exécutent vos tâches dans le service. Une version de plateforme est spécifiée uniquement pour les tâches utilisant le type de lancement Fargate. Si vous ne spécifiez aucune valeur, la dernière version (`LATEST`) est utilisée par défaut.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

AWS Les versions de la plate-forme Fargate sont utilisées pour faire référence à un environnement d'exécution spécifique pour l'infrastructure de tâches Fargate. Lorsque vous spécifiez `LATEST` comme version de plateforme lors de l'exécution d'une tâche ou de la création d'un service, vous obtenez la version de plateforme la plus récente pour vos tâches. Lorsque vous mettez à l'échelle votre service, ces tâches reçoivent la version de plateforme spécifiée dans le déploiement actuel du service. Pour plus d’informations, consultez Versions de plateforme Fargate pour Amazon ECS.

> **Note**
>
> Les versions de plateforme ne sont pas spécifiées pour les tâches utilisant le type de lancement EC2.

Cluster

`cluster`

Type : chaîne

Obligatoire : non

Nom court ou Amazon Resource Name (ARN) complet du cluster sur lequel exécuter votre service. Si vous ne spécifiez aucun cluster, le cluster `default` est sélectionné.

Nom du service

`serviceName`

Type : chaîne

Obligatoire : oui

Nom de votre service. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits d'union et traits de soulignement sont autorisés. Dans un cluster, les noms de service doivent être uniques, mais certains services peuvent avoir des noms similaires dans des clusters différents d'une même région ou de plusieurs régions.

Stratégie de planification

`schedulingStrategy`

Type : String

Valeurs valides : `REPLICA` | `DAEMON`

Obligatoire : non

Stratégie de planification à utiliser. Si aucune stratégie de planification n'est spécifiée, la stratégie `REPLICA` est utilisée. Pour plus d’informations, consultez Services Amazon ECS.

Deux stratégies de planificateur de service sont disponibles :

- `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraintes de placement des tâches

afin de personnaliser la façon dont les tâches sont placées. Pour plus d'informations, consultez Stratégie de planification de réplica.

- `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraintes de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour plus d'informations, consultez Stratégie de planification de démon.

  > **Note**
  >
  > Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`.

Nombre souhaité

`desiredCount`

Type : Integer

Obligatoire : non

Nombre d'instances de la définition de tâche spécifiée à placer et à exécuter dans votre service.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

Ce paramètre est requis si la stratégie de planification `REPLICA` est utilisée. Si le service utilise la stratégie de planification `DAEMON`, ce paramètre est facultatif.

Dans le cadre de l'autoscaling de service, lorsque vous mettez à jour un service en cours d'exécution avec un `desiredCount` inférieur au nombre de tâches actuellement en cours d'exécution, le service est réduit au `desiredCount` spécifié.

Configuration de déploiement

`deploymentConfiguration`

Type : objet

Obligatoire : non

Paramètres de déploiement facultatifs qui contrôlent le nombre de tâches exécutées durant le déploiement et la commande d'arrêt et de démarrage des tâches.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`maximumPercent`

Type : Integer

Obligatoire : non

Si un service utilise le type de déploiement de mise à jour propagée (`ECS`), le paramètre `maximumPercent` représente une limite supérieure du nombre de tâches de votre service autorisées à avoir l'état `RUNNING`, `STOPPING` ou `PENDING` pendant un déploiement. Il est exprimé en tant que pourcentage de `desiredCount` arrondi à la valeur inférieure la plus proche. Vous pouvez utiliser ce paramètre pour définir la taille des lots de déploiement. Par exemple, si votre service utilise le planificateur de services `REPLICA` et a un `desiredCount` de quatre tâches et une valeur `maximumPercent` de 200 %, le planificateur démarre quatre nouvelles tâches avant d'arrêter les quatre tâches les plus anciennes. Cela s'applique à condition que les ressources de cluster nécessaires à cette opération soient disponibles. La valeur `maximumPercent` par défaut pour un service à l'aide du planificateur de service `REPLICA` est de 200 %.

Le planificateur Amazon ECS utilise ce paramètre pour remplacer les tâches défectueuses en démarrant d'abord les tâches de remplacement, puis en arrêtant les tâches défectueuses, à condition que les ressources du cluster permettant de démarrer les tâches de remplacement soient disponibles. Pour de plus amples informations sur la manière dont le planificateur remplace les tâches défectueuses, consultez la section Services Amazon ECS.

Si votre service utilise le type de planificateur de service `DAEMON`, le `maximumPercent` doit rester à 100 %. C'est la valeur par défaut.

Le nombre maximum de tâches lors d'un déploiement est le `desiredCount` multiplié par le `maximumPercent`/100, arrondi au nombre entier inférieur le plus proche.

Si un service utilise l'un des blue/green (`CODE_DEPLOY`) ou `EXTERNAL` les types de déploiement et les tâches qui utilisent le type de lancement EC2, le pourcentage maximum la valeur est définie sur la valeur par défaut. Cette valeur sert à définir la limite supérieure du

nombre de tâches du service qui restent dans l'état `RUNNING` pendant que les instances de conteneur sont dans l'état `DRAINING`.

> **Note**
>
> Vous ne pouvez pas spécifier de personnalisation `maximumPercent` valeur pour un service qui utilise soit le blue/green (`CODE_DEPLOY`) ou `EXTERNAL` types de déploiement et comporte des tâches qui utilisent l'EC2.

Si le service utilise soit le blue/green (`CODE_DEPLOY`) ou `EXTERNAL` types de déploiement et si les tâches du service utilisent le Fargate, la valeur maximale en pourcentage n'est pas utilisée. La valeur est toujours renvoyée lors de la description de votre service.

`minimumHealthyPercent`

Type : Integer

Obligatoire : non

Si un service utilise le type de déploiement de mise à jour propagée (`ECS`), `minimumHealthyPercent` représente une limite inférieure du nombre de tâches de votre service qui doivent rester l'état `RUNNING` pendant un déploiement. Cela est exprimé en tant que pourcentage de `desiredCount` arrondi à la valeur supérieure la plus proche. Vous pouvez utiliser ce paramètre pour procéder au déploiement sans avoir recours à une capacité de cluster supplémentaire.

Par exemple, si votre service a un `desiredCount` de quatre tâches, un `minimumHealthyPercent` de 50 % et un `maximumPercent` de 100 %, le planificateur de service peut arrêter deux tâches existantes pour libérer de la capacité de cluster avant de lancer deux nouvelles tâches.

Si certaines tâches sont défectueuses et si `maximumPercent` ne permet pas au planificateur Amazon ECS de démarrer des tâches de remplacement, le planificateur arrête les tâches défectueuses une par une, en utilisant `minimumHealthyPercent` comme contrainte, afin de libérer de la capacité pour lancer des tâches de remplacement. Pour de plus amples informations sur la manière dont le planificateur remplace les tâches défectueuses, consultez la section Services Amazon ECS.

Pour les services qui n'utilisent pas d'équilibreur de charge, il convient de noter ce qui suit :

- Un service est considéré comme intègre si tous les conteneurs essentiels dans les tâches du service réussissent leurs surveillances d'état.
- Si une tâche n'a pas de conteneurs essentiels avec une surveillance de l'état définie, le planificateur de service attend 40 secondes une fois qu'une tâche a atteint l'état `RUNNING` avant que celle-ci soit comptée dans le pourcentage minimal total d'intégrité.
- Si une tâche comporte un ou plusieurs conteneurs essentiels avec une surveillance de l'état définie, le planificateur de service attend que la tâche atteigne un état intègre avant de la compter dans le pourcentage minimum total d'intégrité. Une tâche est considérée comme intègre lorsque tous ses conteneurs essentiels ont réussi leurs surveillances d'état. La durée pendant laquelle le planificateur de service peut attendre est déterminée par les paramètres de surveillance de l'état du conteneur. Pour plus d’informations, consultez Surveillance de l’état.

Pour les services qui utilisent un équilibreur de charge, il convient de noter ce qui suit :

- Si une tâche n'a pas de conteneurs essentiels avec une surveillance de l'état définie, le planificateur de services attend que la surveillance de l'état du groupe cible de l'équilibreur de charge retrouve un état intègre avant de compter la tâche dans le pourcentage minimal total d'intégrité.
- Si une tâche possède un conteneur essentiel avec une surveillance de l'état définie, le planificateur de services attend que la tâche atteigne un état intègre et que la surveillance de l'état du groupe cible d'équilibrage de charge retrouve un état intègre avant de compter la tâche dans le pourcentage minimal total d'intégrité.

La valeur par défaut d'un service de réplica pour `minimumHealthyPercent` est 100 %. La valeur par défaut `minimumHealthyPercent` valeur pour un service utilisant `DAEMON` le calendrier d'entretien est de 0 % pour le AWS CLI, le AWS les SDK et les API et 50 % pour le Console de gestion AWS.

Le nombre minimum de tâches saines lors d'un déploiement correspond à `desiredCount` multiplié par `minimumHealthyPercent`/100, arrondi au nombre entier supérieur le plus proche.

Si un service utilise l'un des blue/green (`CODE_DEPLOY`) ou `EXTERNAL` types de déploiement et exécute des tâches qui utilisent l'EC2, le pourcentage de santé minimum la valeur est définie sur la valeur par défaut. Cette valeur sert à définir la limite inférieure du nombre de tâches du service qui restent dans l’état `RUNNING` pendant que les instances de conteneur sont dans l’état `DRAINING`.

> **Note**
>
> Vous ne pouvez pas spécifier de personnalisation `maximumPercent` valeur pour un service qui utilise soit le blue/green (`CODE_DEPLOY`) ou `EXTERNAL` types de déploiement et comporte des tâches qui utilisent l'EC2.

Si un service utilise l'un des blue/green (`CODE_DEPLOY`) ou `EXTERNAL` les types de déploiement et l'exécution de tâches qui utilisent Fargate, la valeur minimale de pourcentage sain n'est pas utilisée, bien qu'elle soit renvoyée lors de la description de votre service.

`earlySuccessCriteria`

Type : objet

Obligatoire : non

Les premiers critères de réussite qui déterminent le moment où Amazon ECS termine un déploiement continu. Grâce à des critères de réussite précoces, Amazon ECS termine le déploiement lorsqu'un pourcentage des tâches que vous définissez est en cours d'exécution et fonctionne correctement sur la révision du service cible. Vous pouvez également configurer si Amazon ECS nettoie les tâches relatives à la révision du service source avant ou après la fin du déploiement. Pour plus d'informations, consultez Terminez les déploiements continus d'Amazon ECS à un stade précoce en fonction de critères de réussite précoces.

`enable`

Type : booléen

Obligatoire : non

Spécifie si le critère de réussite précoce est activé pour le service. La valeur par défaut est `false`.

`healthyPercent`

Type : Integer

Obligatoire : non

Spécifie le pourcentage du nombre souhaité qui doit être en cours d'exécution et en bon état lors de la révision du service cible avant qu'Amazon ECS ne termine le

déploiement, arrondi au chiffre supérieur. La valeur doit être comprise entre le service `minimumHealthyPercent` et 100, inclus. La valeur par défaut est 100.

`sourceServiceRevisionCleanup`

Type : String

Valeurs valides : `BLOCKING` | `DEFERRED`

Obligatoire : non

Détermine à quel moment Amazon ECS nettoie les tâches de la révision du service source. Avec `BLOCKING`, Amazon ECS nettoie les tâches relatives à la révision du service source avant de terminer le déploiement. Avec `DEFERRED`, Amazon ECS termine d'abord le déploiement, puis nettoie les tâches relatives à la révision du service source en dehors du déploiement. La valeur par défaut est `BLOCKING`.

Contrôleur de déploiement

`deploymentController`

Type : objet

Obligatoire : non

Le contrôleur de déploiement à utiliser pour le service. Si aucun contrôleur de déploiement n'est spécifié, le contrôleur `ECS` est utilisé. Pour plus d'informations, consultez Services Amazon ECS.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`type`

Type : String

Valeurs valides : `ECS` | `CODE_DEPLOY` | `EXTERNAL`

Obligatoire : oui

Type de contrôleur de déploiement à utiliser. Il existe trois types de contrôleurs de déploiement disponibles :

`ECS`

Le contrôleur de déploiement Amazon ECS prend en charge plusieurs stratégies de déploiement : mise à jour continue, blue/green, linéaire et canari. Le type de déploiement de mises à jour continues implique le remplacement de la version en cours d'exécution du conteneur par la dernière version. Blue/green les déploiements créent un nouvel environnement et réorientent le trafic en une seule fois. Les déploiements linéaires déplacent progressivement le trafic par incréments de pourcentage égaux. Les déploiements de Canary transfèrent d'abord un petit pourcentage du trafic, puis le trafic restant. Le nombre de conteneurs ajoutés ou supprimés par Amazon ECS du service lors de la propagation des mises à jour est contrôlé en ajustant le nombre minimum et maximum de tâches saines lors d'un déploiement de service autorisé, tel que spécifié dans le deploymentConfiguration.

`CODE_DEPLOY`

Le blue/green (`CODE_DEPLOY`) le type de déploiement utilise blue/green modèle de déploiement alimenté par CodeDeploy, qui vous permet de vérifier un nouveau déploiement d'un service avant de lui envoyer du trafic de production.

`EXTERNAL`

Utilisez le type de déploiement externe lorsque vous souhaitez utiliser n'importe quel contrôleur de déploiement tiers pour contrôler complètement le processus de déploiement d'un service Amazon ECS.

Placement des tâches

`placementConstraints`

Type : tableau d'objets

Obligatoire : non

Tableau d'objets de contraintes de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier un maximum de 10 contraintes par tâche. Cette limite inclut les contraintes de la définition de tâche et celles spécifiées lors de l'exécution. Si vous utilisez Fargate, les contraintes de placement des tâches ne sont pas prises en charge.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d’un nouveau service.

`type`

Type : chaîne

Obligatoire : non

Type de contrainte. Utilisez `distinctInstance` pour vous assurer que chaque tâche d'un groupe particulier s'exécute sur une instance de conteneur différente. Utilisez `memberOf` pour limiter la sélection à un groupe de candidats valides. La valeur `distinctInstance` n'est pas prise en charge dans les définitions de tâche.

`expression`

Type : chaîne

Obligatoire : non

Expression de langage de requête de cluster à appliquer à la contrainte. Vous ne pouvez pas spécifier d'expression si le type de contrainte est `distinctInstance`. Pour plus d'informations, consultez Création d'expressions pour définir des instances de conteneur pour les tâches Amazon ECS.

`placementStrategy`

Type : tableau d'objets

Obligatoire : non

Objets de stratégie de placement à utiliser pour les tâches de votre service. Vous pouvez spécifier jusqu'à quatre règles de stratégie par service.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`type`

Type : String

Valeurs valides : `random` | `spread` | `binpack`

Obligatoire : non

Type de stratégie de placement. La stratégie de placement `random` place les tâches de façon aléatoire sur les candidats disponibles. La stratégie de placement `spread` répartit le placement sur les candidats disponibles de façon uniforme en fonction du paramètre du

`field`. La stratégie `binpack` place les tâches sur les candidats disponibles qui ont la quantité disponible la moins élevée de la ressource spécifiée avec le paramètre `field`. Par exemple, si vous utilisez un BinPack sur la mémoire, une tâche est placée sur l'instance avec la quantité la moins élevée de mémoire restante, mais tout de même suffisante pour exécuter la tâche.

`field`

Type : chaîne

Obligatoire : non

Champ sur lequel appliquer la stratégie de placement. Pour la stratégie de placement `spread`, les valeurs valides sont `instanceId` (ou `host`, qui produit le même effet), ou toute plateforme ou tout attribut personnalisé appliqué à une instance de conteneur, comme `attribute:ecs.availability-zone`. Pour la stratégie de placement `binpack`, les valeurs valides sont `cpu` et `memory`. Pour la stratégie de placement `random`, ce champ n'est pas utilisé.

Étiquettes

`tags`

Type : tableau d'objets

Obligatoire : non

Métadonnées que vous appliquez au service pour faciliter le classement et l'organisation. Chaque étiquette est constituée d'une clé et d'une valeur facultative que vous définissez. Lorsqu'un service est supprimé, les étiquettes sont également supprimées. Un maximum de 50 balises peut être appliqué au service. Pour plus d’informations, consultez Balisage des ressources Amazon ECS.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d’un nouveau service.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Partie d'une paire clé-valeur qui constitue une étiquette. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

Partie facultative d'une paire clé-valeur qui constitue une étiquette. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`enableECSManagedTags`

Type : Boolean

Valeurs valides : `true` | `false`

Obligatoire : non

Spécifie si utiliser les identifications gérées par Amazon ECS pour les tâches dans le service. La valeur par défaut est `false` si aucune valeur n'est spécifiée. Pour plus d'informations, consultez Mise à jour des balises pour la facturation.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE`

Obligatoire : non

Indique s'il faut copier les étiquettes de la définition de tâche ou du service dans les tâches du service. Si aucune valeur n'est spécifiée, les étiquettes ne sont pas copiées. Les étiquettes ne peuvent être copiées dans les tâches du service que lors de la création du service. Pour ajouter des balises à une tâche après la création du service ou de la tâche, utilisez l'action d'API `TagResource`.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

Configuration réseau

`networkConfiguration`

Type : objet

Obligatoire : non

Configuration réseau du service. Ce paramètre est obligatoire pour les définitions de tâches qui utilisent le mode réseau `awsvpc` afin de recevoir leur propre interface réseau Elastic. Il n'est pas pris en charge avec les autres modes réseau. Si vous utilisez Fargate, le mode réseau `awsvpc` est requis. Pour plus d'informations sur la mise en réseau pour EC2, consultez la section Options de mise en réseau des tâches Amazon ECS pour EC2. Pour plus d'informations sur la mise en réseau pour Fargate, consultez la section Options de mise en réseau des tâches Amazon ECS pour Fargate.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

`awsvpcConfiguration`

Type : objet

Obligatoire : non

Objet représentant les sous-réseaux et les groupes de sécurité pour une tâche ou un service.

`subnets`

Type : tableau de chaînes

Obligatoire : oui

Sous-réseaux associés à la tâche ou au service. Il existe une limite de 16 sous-réseaux pouvant être spécifiés selon `awsvpcConfiguration`.

`securityGroups`

Type : tableau de chaînes

Obligatoire : non

Groupes de sécurité associés à la tâche ou au service. Si vous ne spécifiez pas de groupe de sécurité, c'est le groupe de sécurité par défaut du VPC qui est utilisé. Il existe une limite de cinq groupes de sécurité pouvant être spécifiés selon `awsvpcConfiguration`.

`assignPublicIP`

Type : String

Valeurs valides : `ENABLED` | `DISABLED`

Obligatoire : non

Si l'interface réseau Elastic de la tâche reçoit une adresse IP publique ou non. Si aucune valeur n'est spécifiée, la valeur par défaut de `DISABLED` est utilisée.

`healthCheckGracePeriodSeconds`

Type : Integer

Obligatoire : non

La durée, en secondes, pendant laquelle le planificateur de service Amazon ECS ignore les surveillances de l'état défectueuses d'Elastic Load Balancing, de VPC Lattice et des conteneurs après le premier démarrage d'une tâche. Si vous ne spécifiez aucune valeur pour la période de grâce de surveillance de l'état, la valeur par défaut 0 est utilisée. Si vous n'utilisez aucune surveillance de l'état, `healthCheckGracePeriodSeconds` n'est pas utilisé.

Si les tâches de votre service mettent du temps à démarrer et à répondre, vous pouvez spécifier une période de grâce de surveillance de l'état pouvant aller jusqu'à 2 147 483 647 secondes (environ 69 ans). Pendant ce temps, le planificateur du service Amazon ECS ignore le statut de la surveillance de l'état. La période de grâce peut empêcher le planificateur de services de marquer les tâches comme non saines et de les arrêter avant qu'elles aient le temps de s'exécuter.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

`loadBalancers`

Type : tableau d'objets

Obligatoire : non

Un objet d'équilibreur de charge qui représente l'équilibreur de charge à utiliser avec votre service. Pour les services qui utilisent un équilibreur de charge Application Load Balancer ou un Network Load Balancer, il existe une limite de cinq groupes cibles que vous pouvez attacher à un service.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

Une fois que vous avez créé un service, la configuration de l'équilibreur de charge ne peut pas être modifiée à partir de la Console de gestion AWS. Vous pouvez utiliser le AWS Copilote, AWS CloudFormation, AWS CLI ou SDK pour modifier la configuration de l'équilibreur de charge pour ECS contrôleur de déploiement roulant uniquement, non AWS CodeDeploy blue/green ou externe. Lorsque vous ajoutez, mettez à jour ou supprimez une configuration d'équilibreur de charge, Amazon ECS lance un nouveau déploiement avec la configuration mise à jour d'Elastic Load Balancing. Cela entraîne l'enregistrement et le désenregistrement des tâches auprès des équilibreurs de charge. Nous vous recommandons de vérifier cela dans un environnement de test avant de mettre à jour la configuration d'Elastic Load Balancing. Pour plus d'informations sur la façon de modifier la configuration, voir UpdateService dans le Référence de l'API Amazon Elastic Container Service.

Pour les équilibreurs de charge Application Load Balancer et les Network Load Balancer, cet objet doit contenir l'ARN du groupe cible de l'équilibreur de charge, le nom du conteneur (tel qu'il apparaît dans une définition de conteneur) et le port du conteneur pour permettre l'accès à partir de l'équilibreur de charge. Lorsqu'une tâche de ce service est placée sur une instance de conteneur, la combinaison d'instance et de port de conteneur est enregistrée comme cible dans le groupe cible spécifié.

`targetGroupArn`

Type : chaîne

Obligatoire : non

Amazon Resource Name (ARN) complet du groupe cible Elastic Load Balancing associés à un service.

L'ARN d'un groupe cible n'est spécifié que si vous utilisez un Application Load Balancer ou un Network Load Balancer.

`loadBalancerName`

Type : chaîne

Obligatoire : non

Nom de l'équilibreur de charge à associer au service.

Si vous utilisez un Application Load Balancer ou un Network Load Balancer, omettez le paramètre de nom de l'équilibreur de charge.

`containerName`

Type : chaîne

Obligatoire : non

Nom du conteneur (tel qu'il apparaît dans une définition de conteneur) à associer à l'équilibreur de charge.

`containerPort`

Type : Integer

Obligatoire : non

Port du conteneur à associer à l'équilibreur de charge. Ce port doit correspondre à un élément `containerPort` dans la définition de tâche utilisée par les tâches du service. Pour les tâches qui utilisent EC2, l'instance de conteneur doit autoriser le trafic entrant sur le `hostPort` du mappage de port.

`role`

Type : chaîne

Obligatoire : non

Nom abrégé ou ARN complet du rôle IAM qui permet à Amazon ECS d'adresser des appels à votre équilibreur de charge en votre nom. Ce paramètre est uniquement autorisé si vous utilisez un équilibreur de charge avec un seul groupe cible pour votre service, et si votre définition de tâche n'utilise pas le mode réseau `awsvpc`. Si vous spécifiez le paramètre `role`, vous devez également spécifier un objet d'équilibreur de charge avec le paramètre `loadBalancers`.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

Si le rôle spécifié possède un chemin d'accès autre que `/`, vous devez spécifier l'ARN de rôle complet (solution recommandée) ou ajouter ce chemin d'accès comme préfixe du nom de rôle. Par exemple, si un rôle nommé `bar` a le chemin d'accès `/foo/`, vous devez spécifier `/foo/bar` comme nom du rôle. Pour plus d'informations, consultez Noms conviviaux et chemins dans le Guide de l'utilisateur IAM.

> **Important**
>
> Si votre compte a déjà créé le rôle lié à un service Amazon ECS service, ce rôle est utilisé par défaut pour votre service, sauf si vous spécifiez un rôle ici. Le rôle lié à un service est obligatoire si votre définition de tâche utilise le mode réseau awsvpc, auquel cas vous ne devez pas spécifier de rôle ici. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.

`serviceConnectConfiguration`

Type : objet

Obligatoire : non

Cette configuration permet à ce service de découvrir des services et de s'y connecter, et, inversement, elle permet à d'autres services au sein d'un espace de noms de découvrir ce service et de s'y connecter.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

Pour plus d'informations, consultez Utilisation de Service Connect pour connecter des services Amazon ECS avec des noms abrégés.

`enabled`

Type : Boolean

Obligatoire : oui

Spécifie s'il faut utiliser Service Connect avec ce service.

`namespace`

Type : chaîne

Obligatoire : non

Le nom abrégé ou le nom de ressource Amazon complet (ARN) du AWS Cloud Map espace de noms à utiliser avec Service Connect. L'espace de noms doit se trouver dans la même

Région AWS que le service Amazon ECS et le cluster. Le type d'espace de noms n'a aucune incidence sur Service Connect. Pour plus d'informations sur AWS Cloud Map, voir Travailler avec les services dans le AWS Cloud Map Guide du développeur.

`services`

Type : tableau d'objets

Obligatoire : non

Tableau d'objets de service Service Connect. Il s'agit de noms et d'alias (également appelés points de terminaison) utilisés par d'autres services Amazon ECS pour se connecter à ce service.

Ce champ n'est pas obligatoire pour un service Amazon ECS « client » membre d'un espace de noms uniquement pour se connecter à d'autres services au sein de cet espace de noms. Par exemple, une application frontale qui accepte les demandes entrantes provenant d'un équilibreur de charge attaché au service ou par d'autres moyens.

Un objet sélectionne un port à partir de la définition de la tâche, attribue un nom au AWS Cloud Map service, ainsi qu'un ensemble d'alias (également appelés points de terminaison) et de ports permettant aux applications clientes de faire référence à ce service.

`portName`

Type : chaîne

Obligatoire : oui

Le `portName` doit correspondre au `name` de l'un des `portMappings` de tous les conteneurs dans la définition de tâche de ce service Amazon ECS.

`discoveryName`

Type : chaîne

Obligatoire : non

Le `discoveryName` est le nom du nouveau AWS Cloud Map service créé par Amazon ECS pour ce service Amazon ECS. Ce nom doit être unique dans l'espace de noms AWS Cloud Map .

Si ce champ n'est pas spécifié, `portName` est utilisé.

`clientAliases`

Type : tableau d'objets

Obligatoire : non

Liste des alias clients pour ce service Service Connect. Vous les utilisez pour attribuer des noms qui peuvent être utilisés par les applications clientes. Le nombre maximum d'alias clients dont vous pouvez disposer dans cette est de 1.

Chaque alias (« point de terminaison ») est un nom DNS et un numéro de port que les autres services Amazon ECS (« clients ») peuvent utiliser pour se connecter à ce service.

Chaque nom et chaque combinaison de port doivent être uniques dans l'espace de noms.

Ces noms sont configurés dans chaque tâche du service client, et non dans AWS Cloud Map. Les demandes DNS visant à résoudre ces noms ne quittent pas la tâche et ne sont pas prises en compte dans le quota de demandes DNS par seconde par l'interface réseau Elastic.

`port`

Type : Integer

Obligatoire : oui

Numéro de port d'écoute du proxy Service Connect. Ce port est disponible dans toutes les tâches du même espace de noms.

Pour éviter de modifier vos applications dans les services clients Amazon ECS, attribuez-lui le même port que celui que l'application client utilise par défaut.

`dnsName`

Type : chaîne

Obligatoire : non

Le `dnsName` est le nom que vous utilisez dans les applications des tâches client pour vous connecter à ce service. Le nom doit être une étiquette DNS valide.

Si ce champ n'est pas spécifié, la valeur par défaut est `discoveryName.namespace`. Si le `discoveryName` n'est pas spécifié, le `portName` issu de la définition de la tâche est utilisé.

Pour éviter de modifier vos applications dans les services clients Amazon ECS, attribuez-lui le même nom que celui que l'application client utilise par défaut. Par exemple, quelques noms courants sont `database`, db ou le nom en minuscules d'une base de données, comme `mysql` ou `redis`.

`ingressPortOverride`

Type : Integer

Obligatoire : non

(Facultatif) Numéro de port que le proxy Service Connect utilise pour écouter.

Utilisez la valeur de ce champ pour contourner le proxy pour le trafic sur le numéro de port spécifié dans le `portMapping` nommé de la définition de tâche de cette application, puis utilisez-la dans les groupes de sécurité de votre Amazon VPC pour autoriser le trafic vers le proxy de ce service Amazon ECS.

En mode `awsvpc`, la valeur par défaut est le numéro du port du conteneur spécifié dans le `portMapping` nommé dans la définition de tâche de cette application. En mode `bridge`, la valeur par défaut correspond au port éphémère du proxy Service Connect.

`logConfiguration`

Tapez : LogConfiguration Objet

Obligatoire : non

Cela définit l'endroit où les journaux du proxy Service Connect sont publiés. Utilisez les journaux pour le débogage lors d'événements inattendus. Cette configuration définit le paramètre `logConfiguration` dans le conteneur de proxy Service Connect pour chaque tâche de ce service Amazon ECS. Le conteneur de proxy n'est pas spécifié dans la définition de tâche.

Nous vous recommandons d'utiliser la même configuration de journal que les conteneurs d'applications de la définition de tâche pour ce service Amazon ECS. Pour FireLens, il s'agit de la configuration du journal du conteneur d'applications. Ce n'est pas le FireLens conteneur log router qui utilise le `fluent-bit` ou `fluentd` image du conteneur.

`accessLogConfiguration`

Tapez : ServiceConnectAccessLogConfiguration Objet

Obligatoire : non

La configuration de la journalisation des accès à Service Connect, y compris le format du journal et la question de savoir si les journaux doivent inclure des chaînes de requête. Les journaux d'accès capturent des informations détaillées sur les demandes adressées à votre service, notamment les modèles de demandes, le code de réponse et les données de synchronisation. Pour activer les journaux d'accès, vous devez également spécifier `logConfiguration` dans le `serviceConnectConfiguration`.

`serviceRegistries`

Type : tableau d'objets

Obligatoire : non

Détails de la configuration de la découverte de service pour votre service. Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

`registryArn`

Type : chaîne

Obligatoire : non

Amazon Resource Name (ARN) du registre de service. Le registre de services actuellement pris en charge est AWS Cloud Map. Pour plus d'informations, consultez Utilisation des services dans le Guide du développeur AWS Cloud Map .

`port`

Type : Integer

Obligatoire : non

Valeur de port utilisée si votre service de découverte de service a spécifié un registre SRV. Ce champ est obligatoire si le mode réseau `awsvpc` et les registres SRV sont utilisés.

`containerName`

Type : chaîne

Obligatoire : non

La valeur du nom du conteneur, déjà spécifiée dans la définition de tâche, à utiliser pour votre service de découverte de service. Cette valeur est spécifiée dans la définition de tâche. Si la définition de tâche que votre tâche de service spécifie utilise le mode réseau `bridge` ou `host`, vous devez spécifier une combinaison `containerName` et `containerPort` à partir de la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc` et qu'un registre DNS de type SRV est utilisé, vous devez spécifier une combinaison `containerName` et `containerPort` ou une valeur `port`, mais pas les deux.

`containerPort`

Type : Integer

Obligatoire : non

La valeur du port à utiliser pour votre service de découverte de service. Cette valeur est spécifiée dans la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `bridge` ou `host`, vous devez spécifier une combinaison `containerName` et `containerPort` à partir de la définition de tâche. Si la définition de tâche spécifiée par votre tâche de service utilise le mode réseau `awsvpc` et qu'un registre DNS de type SRV est utilisé, vous devez spécifier une combinaison `containerName` et `containerPort` ou une valeur `port`, mais pas les deux.

Jeton client

`clientToken`

Type : chaîne

Obligatoire : non

L'identifiant unique, sensible à la casse, que vous devez fournir afin de garantir l'idempotence de la demande. Il peut comporter jusqu'à 32 ASCII caractères.

Rééquilibrage des zones de disponibilité

`availabilityZoneRebalancing`

Type : chaîne

Obligatoire : non

Indique si le service utilise le rééquilibrage des zones de disponibilité. Les valeurs valides sont `ENABLED` et `DISABLED`.

Lorsque vous mettez à jour un service, ce paramètre ne déclenche pas le déploiement d'un nouveau service.

Comportement par défaut :

- Pour les nouveaux services : si aucune valeur n'est spécifiée, la valeur par défaut est `DISABLED`.
- Pour les services existants : si aucune valeur n'est spécifiée, Amazon ECS définit par défaut la valeur existante. Si aucune valeur n'a été définie précédemment, Amazon ECS définit la valeur sur `DISABLED`.

Pour plus d'informations sur les zones de disponibilité, consultez la section Équilibrage d'un service Amazon ECS entre les zones de disponibilité.

Configurations de volume

`volumeConfigurations`

Type : objet

Obligatoire : non

La configuration qui sera utilisée pour créer des volumes pour les tâches gérées par le service. Seuls les volumes marqués comme `configuredAtLaunch` dans la définition de tâche peuvent être configurés à l'aide de cet objet.

Lorsque vous mettez à jour un service, ce paramètre déclenche un nouveau déploiement de service.

Cet objet est requis pour associer des volumes Amazon EBS à des tâches gérées par un service. Pour plus d'informations, consultez Utilisation des volumes Amazon EBS avec Amazon ECS.

`name`

Type : chaîne

Obligatoire : oui

Le nom d'un volume configuré lors de la création ou de la mise à jour d'un service. Jusqu'à 255 lettres (majuscules et minuscules), chiffres, traits de soulignement (_) et traits d'union (-) sont autorisés. Cette valeur doit correspondre au nom du volume spécifié dans la définition de tâche.

`managedEBSVolume`

Type : objet

Obligatoire : non

La configuration de volume utilisée pour créer des volumes Amazon EBS attachés à des tâches gérées par un service lors de sa création ou de sa mise à jour. Un seul volume est joint par tâche.

`encrypted`

Type : booléen

Obligatoire : non

Valeurs valides : `true`|`false`

Spécifie s'il faut activer le chiffrement sur chaque volume Amazon EBS créé. Si vous avez activé le chiffrement Amazon EBS par défaut pour un Région AWS pour votre Compte AWS mais définissez ce paramètre sur `false`, ce paramètre sera remplacé et les volumes seront chiffrés avec la clé KMS spécifiée pour le chiffrement par défaut. Pour plus d'informations sur le chiffrement Amazon EBS par défaut, consultez la section Activation du chiffrement Amazon EBS par défaut dans le Guide de l'utilisateur Amazon EBS. Pour plus d'informations sur le chiffrement des volumes Amazon EBS associés aux tâches Amazon ECS, consultez la section Chiffrement des données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS.

`kmsKeyId`

Type : chaîne

Obligatoire : non

L'identifiant du AWS Key Management Service (AWS KMS) clé à utiliser pour le chiffrement Amazon EBS. Si `kmsKeyId` est spécifié, l'état chiffré doit être `true`.

La clé spécifiée à l'aide de ce paramètre remplace la clé par défaut d'Amazon EBS ou toute clé KMS au niveau du cluster pour le chiffrement du stockage géré Amazon ECS que

vous avez éventuellement spécifiée. Pour plus d'informations, consultez Chiffrement des données stockées dans les volumes Amazon EBS associés aux tâches Amazon ECS.

Vous pouvez spécifier la clé KMS en utilisant l'un des éléments suivants :

- ID de clé : par exemple, `1234abcd-12ab-34cd-56ef-1234567890ab`.
- Alias de clé : par exemple, `alias/ExampleAlias`.
- ARN de clé : par exemple, `arn:aws:kms:us-east-1:012345678910:key/1234abcd-12ab-34cd-56ef-1234567890ab`.
- ARN d'alias : par exemple, `arn:aws:kms:us-east-1:012345678910:alias/ExampleAlias`.

> **Important**
>
> AWS authentifie la clé KMS de manière asynchrone. Par conséquent, si vous spécifiez un ID, un alias ou un ARN qui n'est pas valide, l'action peut sembler réussir, mais elle finira par échouer. Pour plus d'informations, consultez la section Résolution des problèmes liés à l'attachement de volume Amazon EBS.

`volumeType`

Type : chaîne

Obligatoire : non

Valeurs valides : `gp2|gp3|io1|io2|sc1|st1|standard`

Le type de volume Amazon EBS. Pour plus d'informations sur les types de volumes, consultez la section Types de volumes Amazon EBS dans le Guide de l'utilisateur Amazon EBS. Le type de volume par défaut est `gp3`.

> **Note**
>
> Le type de volume `standard` n'est pas pris en charge pour les tâches Fargate.

`sizeInGiB`

Type : Integer

Obligatoire : non

Plage valide : entiers compris entre 1 et 16 384.

La taille du volume EBS en gibioctets (Gio). Si vous ne fournissez pas d'ID d'instantané pour configurer un volume à joindre, vous devez fournir une valeur de taille. Si vous configurez un volume pour l'attachement à l'aide d'un instantané, la valeur par défaut est la taille de l'instantané. Vous pouvez ensuite spécifier une taille supérieure ou égale à la taille de l'instantané.

Pour les types de volume `gp2` et `gp3`, la plage valide est comprise entre 1 et 16 384.

Pour les types de volume `io1` et `io2`, la plage valide est comprise entre 4 et 16 384.

Pour les types de volume `st1` et `sc1`, la plage valide est comprise entre 125 et 16 384.

Pour les types de volume `standard`, la plage valide est comprise entre 1 et 1 024.

`snapshotId`

Type : chaîne

Obligatoire : non

L'ID de l'instantané d'un volume Amazon EBS existant qu'Amazon ECS utilise pour créer de nouveaux volumes à attacher. Vous devez spécifier `snapshotId` ou `sizeInGiB`.

`volumeInitializationRate`

Type : Integer

Obligatoire : non

Le taux, en MiB/s, au cours de laquelle les données sont extraites d'un instantané d'un volume Amazon EBS existant afin de créer de nouveaux volumes à joindre. Cette propriété ne peut être spécifiée que si vous spécifiez un `snapshotId`. Pour plus d'informations sur ce taux d'initialisation de volume, y compris la gamme de taux pris en charge pour l'initialisation, consultez la section Initialisation des volumes Amazon EBS dans le Guide de l'utilisateur Amazon EBS.

`iops`

Type : Integer

Obligatoire : non

Le nombre de I/O opérations par seconde (IOPS). Pour les volumes `gp3`, `io1` et `io2`, cela représente le nombre d'IOPS provisionnés pour le volume. Pour `gp2` volumes, cette valeur représente la performance de base du volume et la vitesse à laquelle le volume s'accumule I/O crédits pour bursting. Ce paramètre est requis pour les volumes `io1` et `io2`. Ce paramètre n'est pas pris en charge pour les volumes `gp2`, `st1`, `sc1` ou `standard`.

Pour les volumes `gp3`, la plage de valeurs valide est comprise entre 3 000 et 16 000.

Pour les volumes `io1`, la plage de valeurs valide est comprise entre 100 et 64 000.

Pour les volumes `io2`, la plage de valeurs valide est comprise entre 100 et 64 000.

`throughput`

Type : Integer

Obligatoire : non

Le débit à allouer pour les volumes associés aux tâches gérées par un service.

> **Important**
>
> Ce paramètre n'est pris en charge que pour les volumes `gp3`.

`roleArn`

Type : chaîne

Obligatoire : oui

L'ARN Amazon Resource (ARN) de l'infrastructure Gestion des identités et des accès AWS Rôle (IAM) qui fournit les autorisations Amazon ECS pour gérer les ressources Amazon EBS pour vos tâches. Pour plus d'informations, consultez Rôle IAM d'infrastructure Amazon ECS.

`tagSpecifications`

Type : objet

Obligatoire : non

La spécification des balises à appliquer à chaque volume Amazon EBS.

`resourceType`

Type : chaîne

Obligatoire : oui

Valeurs valides : `volume`

Le type de ressource à baliser à la création.

`tags`

Type : tableau d'objets

Obligatoire : non

Les métadonnées que vous appliquez aux volumes pour vous aider à les classer et à les organiser. Chaque balise est constituée d'une clé et d'une valeur facultative que vous définissez. `AmazonECSCreated` et `AmazonECSManaged` sont des balises réservées ajoutées par Amazon ECS en votre nom. Vous pouvez donc spécifier vous-même un maximum de 48 balises. Lorsqu'un service est supprimé, les balises sont également supprimées. Pour plus d'informations, consultez Balisage des ressources Amazon ECS.

`key`

Type : String

Contraintes de longueur : Longueur minimum de 1. Longueur maximale de 128.

Obligatoire : non

Une des parties d'une paire clé-valeur qui compose une balise. Une clé est une étiquette générale qui fait office de catégorie pour les valeurs d'étiquette plus spécifiques.

`value`

Type : String

Contraintes de longueur : Longueur minimum de 0. Longueur maximale de 256.

Obligatoire : non

La partie facultative d'une paire clé-valeur qui compose une balise. Une valeur agit comme un descripteur au sein d'une catégorie d'étiquette (clé).

`propagateTags`

Type : String

Valeurs valides : `TASK_DEFINITION` | `SERVICE` | `NONE`

Obligatoire : non

Spécifie s'il faut copier les balises de la définition de tâche ou du service vers un volume. Si `NONE` est spécifié ou si aucune valeur n'est spécifiée, les balises ne sont pas copiées.

`fileSystemType`

Type : chaîne

Obligatoire : non

Valeurs valides : `xfs`|`ext3`|`ext4`|`NTFS`

Le type de système de fichiers sur un volume. Le type de système de fichiers du volume détermine le mode de stockage et de récupération des données dans le volume. Pour les volumes créés à partir d'un instantané, vous devez spécifier le même type de système de fichiers que celui utilisé par le volume lors de la création de l'instantané. Si le type de système de fichiers ne correspond pas, la tâche ne démarre pas.

Les valeurs valides pour Linux sont `xfs` ext3, `and` `ext4`. La valeur par défaut pour les volumes attachés à des tâches Linux est `XFS`.

La valeur valide pour Windows est `NTFS`. La valeur par défaut pour les volumes attachés à des tâches Windows est `NTFS`.

Modèle de définition de service

Voici une illustration de la représentation JSON d'une définition de service Amazon ECS service.

EC2

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "EC2",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
```

```
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementConstraints": [
        {
            "type": "distinctInstance",
            "expression": ""
        }
    ],
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
```

```
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
```

```
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Fargate

```
{
    "cluster": "",
    "serviceName": "",
    "taskDefinition": "",
    "loadBalancers": [
        {
            "targetGroupArn": "",
            "loadBalancerName": "",
            "containerName": "",
            "containerPort": 0
        }
    ],
    "serviceRegistries": [
        {
            "registryArn": "",
            "port": 0,
            "containerName": "",
            "containerPort": 0
        }
    ],
    "desiredCount": 0,
    "clientToken": "",
    "launchType": "FARGATE",
    "capacityProviderStrategy": [
        {
            "capacityProvider": "",
```

```
            "weight": 0,
            "base": 0
        }
    ],
    "platformVersion": "",
    "platformFamily": "",
    "role": "",
    "deploymentConfiguration": {
        "deploymentCircuitBreaker": {
            "enable": true,
            "rollback": true
        },
        "maximumPercent": 0,
        "minimumHealthyPercent": 0,
        "alarms": {
            "alarmNames": [
                ""
            ],
            "enable": true,
            "rollback": true
        }
    },
    "placementStrategy": [
        {
            "type": "binpack",
            "field": ""
        }
    ],
    "networkConfiguration": {
        "awsvpcConfiguration": {
            "subnets": [
                ""
            ],
            "securityGroups": [
                ""
            ],
            "assignPublicIp": "DISABLED"
        }
    },
    "healthCheckGracePeriodSeconds": 0,
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "EXTERNAL"
    },
```

```
    "tags": [
        {
            "key": "",
            "value": ""
        }
    ],
    "enableECSManagedTags": true,
    "propagateTags": "TASK_DEFINITION",
    "enableExecuteCommand": true,
    "availabilityZoneRebalancing": "ENABLED",
    "serviceConnectConfiguration": {
        "enabled": true,
        "namespace": "",
        "services": [
            {
                "portName": "",
                "discoveryName": "",
                "clientAliases": [
                    {
                        "port": 0,
                        "dnsName": ""
                    }
                ],
                "ingressPortOverride": 0
            }
        ],
        "logConfiguration": {
            "logDriver": "journald",
            "options": {
                "KeyName": ""
            },
            "secretOptions": [
                {
                    "name": "",
                    "valueFrom": ""
                }
            ]
        }
    },
    "volumeConfigurations": [
        {
            "name": "",
            "managedEBSVolume": {
                "encrypted": true,
```

```
                "kmsKeyId": "",
                "volumeType": "",
                "sizeInGiB": 0,
                "snapshotId": "",
                "volumeInitializationRate": 0,
                "iops": 0,
                "throughput": 0,
                "tagSpecifications": [
                    {
                        "resourceType": "volume",
                        "tags": [
                            {
                                "key": "",
                                "value": ""
                            }
                        ],
                        "propagateTags": "NONE"
                    }
                ],
                "roleArn": "",
                "filesystemType": ""
            }
        }
    ]
}
```

Vous pouvez créer ce modèle de définition de service à l'aide de la commande AWS CLI suivante.

```
aws ecs create-service --generate-cli-skeleton
```

Balisage des ressources Amazon ECS

Pour vous aider à gérer vos ressources Amazon ECS, vous pouvez éventuellement affecter vos propres métadonnées à chaque ressource à l'aide de balises. Chaque balise est constituée d'une clé et d'une valeur facultative.

Vous pouvez utiliser des étiquettes pour classer vos ressources Amazon ECS de différentes manières, par exemple, par objectif, par propriétaire ou par environnement. Cela est utile lorsque vous avez de nombreuses ressources du même type. Vous pouvez identifier rapidement une ressource spécifique en fonction des étiquettes que vous lui avez attribuées. Par exemple, vous pouvez définir pour les instances de votre compte Amazon ECS un ensemble d'étiquettes. Cela vous aide à suivre le propriétaire et le niveau de la pile de chaque instance.

Vous pouvez utiliser des balises pour vos Rapports d'utilisation et de coût. Vous pouvez utiliser ces rapports pour analyser le coût et de l'utilisation de vos ressources Amazon ECS. Pour de plus amples informations, veuillez consulter the section called “Rapports d'utilisation”.

> **Warning**
>
> Il existe de nombreuses API qui renvoient les clés de balise et leurs valeurs. Le fait de refuser l’accès à `DescribeTags` ne refuse pas automatiquement l’accès aux balises renvoyées par d’autres API. Nous vous recommandons de ne pas inclure de données sensibles dans vos balises.

Nous vous recommandons de concevoir un ensemble de clés d’étiquette répondant à vos besoins pour chaque type de ressource. Vous pouvez utiliser un ensemble de clés de balise cohérent facilite la gestion de vos ressources. Vous pouvez rechercher et filtrer les ressources en fonction des étiquettes que vous ajoutez.

Les balises n'ont pas de signification sémantique pour Amazon ECS et sont interprétées strictement comme des chaînes de caractères. Vous pouvez modifier les clés et valeurs de balise, et vous pouvez retirer des balises d’une ressource à tout moment. Vous pouvez définir la valeur d'une étiquette sur une chaîne vide, mais vous ne pouvez pas définir la valeur d'une étiquette sur null. Si vous ajoutez une balise ayant la même clé qu'une balise existante sur cette ressource, la nouvelle valeur remplace l'ancienne valeur. Lorsque vous supprimez une ressource, toutes les balises associées à cette ressource sont également supprimées.

Si vous utilisez Gestion des identités et des accès AWS (IAM), vous pouvez contrôler quels utilisateurs de votre AWS compte sont autorisés à gérer les tags.

Comment les ressources sont étiquetées

Les tâches, les services, les définitions de tâches Amazon ECS sont balisés de différentes manières :

- Un utilisateur balise manuellement une ressource à l' Console de gestion AWS aide de l'API Amazon ECS AWS CLI, du ou d'un AWS SDK.
- Un utilisateur crée un service ou exécute une tâche autonome et sélectionne l'option Amazon ECS-managed tags.

  Amazon ECS étiquette automatiquement toutes les tâches récemment lancées. Pour de plus amples informations, veuillez consulter the section called “ECS-managed Balises Amazon”.
- Un utilisateur crée une ressource à l'aide de la console. La console étiquette automatiquement les ressources.

  Ces balises sont renvoyées dans AWS CLI les réponses du AWS SDK et sont affichées dans la console. Vous ne pouvez pas modifier ni supprimer ces balises.

  Pour plus d'informations sur les balises ajoutées, consultez la colonne Balises ajoutées automatiquement par la console dans le tableau des ressources relatives à la prise en charge du balisage pour Amazon ECS.

Si vous spécifiez des balises lorsque vous créez une ressource et que les balises ne peuvent pas être appliquées, Amazon ECS annule le processus de création. Cela permet de s'assurer que les ressources sont créées avec des étiquettes ou qu'elles ne sont pas créées du tout, et qu'aucune ressource ne demeurent sans étiquette à tout moment. En attribuant des étiquettes aux ressources au moment de la création, vous pouvez supprimer la nécessité d'exécuter des scripts d'identification personnalisés après la création de ressources.

Le tableau suivant décrit les ressources Amazon ECS qui prennent en charge les balises.

Ressource	Prend en charge les étiquettes	Prend en charge la propagation des étiquettes	Balises ajoutées automatiquement par la console
Tâches Amazon ECS	Oui	Oui, à partir de la définition de tâche.	Clé : `aws:ecs:clusterName` Value (Valeur) : `cluster-name`
Services Amazon ECS	Oui	Oui, à partir soit de la définition de tâche ou de service pour les tâches dans le service.	Clé : `ecs:service:stackId` Value (Valeur) : `arn:aws:cloudformation:` *`arn`*
Ensembles de tâches Amazon ECS	Oui	Non	N/A
Définitions de tâche Amazon ECS	Oui	Non	Clé : `ecs:taskDefinition:createdFrom` Value (Valeur) : `ecs-console-v2`
Clusters Amazon ECS	Oui	Non	Clé : `aws:cloudformation:logical-id` Value (Valeur) : `ECSCluster` Clé : `aws:cloudformation:stack-id`

Ressource	Prend en charge les étiquettes	Prend en charge la propagation des étiquettes	Balises ajoutées automatiquement par la console
			Value (Valeur) : `arn:aws:cloudformation: arn` Clé : `aws:cloudformation:stack-name` Value (Valeur) : `ECS-Console-V2-Cluster- EXAMPLE`
Instances de conteneur Amazon ECS	Oui	Oui, à partir de l'instance Amazon EC2. Pour de plus amples informations, veuillez consulter Ajout de balises à une instance de conteneur Amazon EC2 pour Amazon ECS.	N/A
Instances Amazon ECS externes	Oui	Non	N/A

Ressource	Prend en charge les étiquettes	Prend en charge la propagation des étiquettes	Balises ajoutées automatiquement par la console
Fournisseur de capacité Amazon ECS	Oui. Vous ne pouvez pas étiqueter les fournisseurs prédéfinis de capacité `FARGATE` et `FARGATE_SPOT` .	Oui, uniquement auprès des fournisseurs de capacité pour les instances gérées Amazon ECS. Vous pouvez propager des balises depuis le fournisseur de capacité pour les instances gérées Amazon ECS vers toutes les ressources gérées par le fournisseur, telles que les instances Amazon EC2, le modèle de lancement, les interfaces réseau Elastic et les volumes.	N/A

Balisage des ressources à la création

Les ressources suivantes prennent en charge le balisage lors de la création à l'aide de l'API ou du AWS SDK Amazon ECS : AWS CLI

- Tâches Amazon ECS
- Services Amazon ECS
- Définition de tâche Amazon ECS
- Ensembles de tâches Amazon ECS
- Clusters Amazon ECS
- Instances de conteneur Amazon ECS

- Fournisseurs de capacité Amazon ECS

Amazon ECS a la possibilité d'utiliser l'autorisation de balisage pour la création de ressources. Lorsque l' Compte AWS est configuré pour l'autorisation de balisage, les utilisateurs doivent être autorisés à effectuer les actions qui créent la ressource, par exemple `ecsCreateCluster`. Si vous spécifiez des balises dans l'action de création de ressources, AWS effectue une autorisation supplémentaire pour vérifier si les utilisateurs ou les rôles sont autorisés à créer des balises. Par conséquent, vous devez octroyer des autorisations explicites d'utiliser l'action `ecs:TagResource`. Pour de plus amples informations, veuillez consulter the section called “Baliser des ressources pendant la création”. Pour plus d'informations sur la configuration de cette option, consultez la section the section called “Autorisation de balisage”.

Restrictions

Les restrictions suivantes s'appliquent aux balises :

- Un maximum de 50 balises peut être associé à une ressource.
- Les clés de balises ne peuvent pas être répétées pour une même ressource. Chaque clé de balise doit être unique et ne peut avoir qu'une seule valeur.
- Les touches peuvent comporter jusqu'à 128 caractères UTF-8.
- Les valeurs peuvent comporter jusqu'à 256 caractères UTF-8.
- Si plusieurs Services AWS ressources utilisent votre schéma de balisage, limitez les types de caractères que vous utilisez. Certains services peuvent présenter des restrictions sur les caractères autorisés. Les caractères généralement autorisés sont les lettres, les chiffres, les espaces et les caractères suivants : **`+ - = . _ : / @`**.
- Les clés et les valeurs des balises distinguent les majuscules et minuscules.
- Vous ne pouvez pas utiliser `aws:`, `AWS:` ou n'importe quelle combinaison de majuscules ou minuscules de ce préfixe pour des clés ou des valeurs Ils sont réservés uniquement à AWS l'usage. Vous ne pouvez pas modifier ni supprimer des clés ou valeurs d'étiquette ayant ce préfixe. Les balises avec ce préfixe ne sont pas comptabilisées comme vos balises pour la limite de ressources.

ECS-managed Balises Amazon

Lorsque vous utilisez des ECS-managed balises Amazon, Amazon ECS étiquette automatiquement toutes les tâches récemment lancées et tous les volumes Amazon EBS attachés aux tâches avec les informations du cluster et soit les balises de définition de tâche ajoutées par l'utilisateur, soit les balises de service. Les balises ajoutées sont décrites ci-dessous :

- Tâches autonomes : une balise avec une clé en tant que `aws:ecs:clusterName` et une valeur définies en fonction du nom du cluster. Toutes les balises de définition de tâches qui ont été ajoutées par les utilisateurs. Un volume Amazon EBS associé à une tâche autonome recevra la balise avec une clé définie sur `aws:ecs:clusterName` et une valeur définie sur sur le nom du cluster. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Balisage des volumes Amazon EBS.
- Tâches faisant partie d'un service : une balise avec une clé en tant que `aws:ecs:clusterName` et une valeur définies en fonction du nom du cluster. Balise avec une clé en tant que `aws:ecs:serviceName` et une valeur définies en fonction du nom du service. Balises de l'une des ressources suivantes.
  - Définitions de tâches : toutes les balises de définition de tâches qui ont été ajoutées par les utilisateurs.
  - Services : toutes les balises de service qui ont été ajoutées par les utilisateurs.

    Un volume Amazon EBS associé à une tâche faisant partie d'un service recevra une balise avec une clé définie sur `aws:ecs:clusterName` et une valeur définie sur le nom du cluster, et une balise avec une clé `aws:ecs:serviceName` avec une valeur définie sur le nom du service. Pour plus d'informations sur le balisage des volumes Amazon EBS, consultez la section Balisage des volumes Amazon EBS.

Les options suivantes sont requises pour cette fonctionnalité :

- Vous devez vous inscrire au nouveaux formats Amazon Resource Name (ARN) et identifiants de ressource (ID). Pour de plus amples informations, veuillez consulter Amazon Resource Names (ARN) et ID.
- Lorsque vous utilisez les API pour créer un service ou exécuter une tâche, vous devez définir `enableECSManagedTags` sur `true` pour `run-task` et `create-service`. Pour plus d'informations, consultez create-service et run-task dans la Référence API d'AWS Command Line Interface .

- Amazon ECS utilise des balises gérées pour déterminer à quel moment certaines fonctionnalités sont activées, par exemple l'autoscaling de cluster. Nous vous recommandons de ne pas modifier manuellement les balises afin qu'Amazon ECS puisse gérer efficacement les fonctionnalités.

Mise à jour des balises pour la facturation

AWS fournit un outil de reporting appelé Cost Explorer que vous pouvez utiliser pour analyser le coût et l'utilisation de vos ressources Amazon ECS.

Vous pouvez utiliser Cost Explorer pour afficher les graphiques de votre utilisation et de vos coûts. Vous pouvez afficher les données des 13 derniers mois et prévoir vos dépenses pour les trois prochains mois. Vous pouvez utiliser Cost Explorer pour afficher des schémas de vos dépenses en ressources AWS au fil du temps. Par exemple, vous pouvez l'utiliser pour identifier les zones qui méritent d'être approfondies et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également spécifier des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Vous pouvez utiliser des ECS-managed tags Amazon ou des tags ajoutés par l'utilisateur pour votre rapport sur les coûts et l'utilisation. Pour de plus amples informations, veuillez consulter Rapports d'utilisation d'Amazon ECS.

Pour voir le coût de vos ressources combinées, vous pouvez organiser vos informations de facturation en fonction des ressources possédant les mêmes valeurs de clé d'étiquette. Par exemple, vous pouvez étiqueter plusieurs ressources avec un nom d'application spécifique, puis organiser vos informations de facturation pour afficher le coût total de cette application dans plusieurs services. Pour en savoir plus sur la configuration d'un rapport de répartition des coûts avec des étiquettes, consultez Rapport d'allocation des coûts mensuel dans le guide de l'utilisateur AWS Billing .

En outre, vous pouvez activer les données de répartition des coûts fractionnés pour obtenir des données d'utilisation de la mémoire et du processeur au niveau des tâches dans vos rapports d'utilisation et de coût. Pour de plus amples informations, veuillez consulter Task-level Rapports sur les coûts et l'utilisation.

> **Note**
>
> Si vous avez activé la création de rapports, 24 heures peuvent être nécessaires avant que les données du mois en cours puissent être consultées.

Ajout de balises aux ressources Amazon ECS

Vous pouvez étiqueter des tâches, services, définitions de tâches et clusters nouveaux ou existants. Pour plus d'informations sur l'identification des instances de conteneur, consultez la section Ajout de balises à une instance de conteneur Amazon EC2 pour Amazon ECS.

> **Warning**
>
> N'ajoutez pas de données d'identification personnelle (PII) ou d'autres informations confidentielles ou sensibles dans les étiquettes. Les tags sont accessibles à de nombreux AWS services, y compris la facturation. Les étiquettes ne sont pas destinées à être utilisées pour des données privées ou sensibles.

Vous pouvez utiliser les ressources suivantes pour spécifier des balises lorsque vous créez la ressource.

Sous-tâche	Console	AWS CLI	Action d'API
Exécuter une ou plusieurs tâches.	Exécution d'une application en tant que tâche Amazon ECS	run-task	RunTask
Créer un service.	Création d'un déploiement de mise à jour propagée Amazon ECS	create-service	CreateService
Créer un ensemble de tâches.	Déploiement de services Amazon ECS à l'aide d'un contrôleur tiers	create-task-set	CreateTaskSet
Enregistrer une définition de tâche.	the section called "Création	register-task-definition	RegisterTaskDefinition

Sous-tâche	Console	AWS CLI	Action d'API
	d'une définition de tâche à l'aide de la console”		
Créer un cluster.	Création d’un cluster Amazon ECS pour les charges de travail Fargate	create-cluster	CreateCluster
Exécuter une ou plusieurs instances de conteneur.	Lancement d'une instance de conteneur Amazon ECS Linux	run-instances	RunInstances
Créez un fournisseur de capacité pour les instances gérées Amazon ECS.	Création d’un fournisseur de capacité pour les instances gérées Amazon ECS	créer un fournisseur de capacité	CreateCapacityProvider

Ajout de balises aux ressources existantes (console Amazon ECS)

Vous pouvez ajouter ou supprimer des balises associées à vos clusters, services, tâches et définitions de tâches directement depuis la page de la ressource.

Pour modifier une balise à une ressource individuelle

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans la barre de navigation, sélectionnez le Région AWS à utiliser.
3. Dans le panneau de navigation, sélectionnez un type de ressource (par exemple, Clusters).
4. Sélectionnez la ressource à partir de la liste des ressources et choisissez l'onglet Tags (Balises), puis Manage Tags (Gérer les balises).

5. Configurez vos balises.

   [Ajouter une balise] Choisissez Add tag (Ajouter une balise), puis procédez comme suit :

   - Pour Clé, saisissez le nom de la clé.
   - Pour Valeur, saisissez la valeur de clé.

6. Choisissez Enregistrer.

Ajout de balises à une ressource existante (AWS CLI)

Vous pouvez ajouter ou remplacer une ou plusieurs balises à l'aide de l'API AWS CLI ou d'une API.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

- AWS CLI - tag-ressource
- Action de l'API - TagResource

Balises pour les instances gérées Amazon ECS

Les instances gérées Amazon ECS prennent en charge une combinaison de balises personnalisées et de balises ajoutées AWS qui peuvent être utilisées pour optimiser les coûts. Pour plus d'informations sur l'utilisation des balises de facturation, consultez la section Mise à jour des balises pour la facturation.

Tags ajoutés par AWS

AWS ajoute les balises suivantes à chaque instance gérée Amazon ECS créée par le fournisseur de capacité :

- Amazon ECS ajoute automatiquement les balises réservées `AmazonECSCreated` et `AmazonECSManaged` aux instances gérées Amazon ECS.

- Amazon ECS ajoute les balises système suivantes à chaque instance :
  - Une balise avec une clé définie sur `aws:ecs:clusterName` et une valeur définie sur le nom du cluster.
  - Une balise avec une clé définie sur `aws:ecs:capacityProviderName` et une valeur définie sur le nom du fournisseur de capacité.
  - Une balise avec une clé définie sur `aws:ecs:containerInstanceId` et une valeur définie sur l'ID d'instance de conteneur pour l'instance gérée Amazon ECS.
- Amazon EC2 ajoute la balise système `aws:ec2:managed-launch` avec la valeur `ecs-managed-instances`. En outre, Amazon EC2 ajoute des balises système indiquant le modèle de lancement utilisé pour créer l'instance gérée, ainsi que la flotte Amazon EC2 dont fait partie l'instance gérée.

Balises personnalisées

Vous pouvez ajouter des balises personnalisées supplémentaires aux instances gérées Amazon ECS en ajoutant des balises au fournisseur de capacité et en activant la propagation des balises à l'aide de la propriété `propagateTags`. Les balises du fournisseur de capacité se propagent aux instances gérées que le fournisseur de capacité lance

L'exemple de définition de fournisseur de capacité suivant montre comment les balises peuvent être spécifiées et propagées à partir du fournisseur de capacité lors de la création du fournisseur de capacité en affectant la `CAPACITY_PROVIDER` valeur à `propagateTags`.

```
{
    "name": "my-cluster-managed-instances-cp",
    "cluster": "my-cluster",
     "tags": [
                {
                "key":"tag_key",
                "value":"tag_value"
                }
            ],
    "managedInstancesProvider": {
        "infrastructureRoleArn": "arn:aws:iam::123456789012:role/
ecsInfrastructureRole",
        "propagateTags": "CAPACITY_PROVIDER",
        "instanceLaunchTemplate": {
            "ec2InstanceProfileArn": "arn:aws:iam::123456789012:instance-profile/
ecsInstanceProfile",
```

```
            "networkConfiguration": {
                "subnets": [
                    "subnet-abcdef01234567",
                    "subnet-bcdefa98765432"
                ],
                "securityGroups": [
                    "sg-0123456789abcdef"
                ]
            }
        }
    }
}
```

> **Note**
>
> - Lorsque vous ajoutez de nouvelles balises à un fournisseur de capacité, les balises nouvellement ajoutées ne sont pas propagées aux instances existantes, mais à toutes les instances nouvellement créées.
> - Les clients sont autorisés à ajouter des balises directement aux instances EC2, mais ces balises ne sont pas retransmises au fournisseur de capacité
> - Vous pouvez également étiqueter ou détaguer le fournisseur de capacité à l'aide Ajout de balises à une ressource existante (AWS CLI) de ou Ajout de balises aux ressources existantes (console Amazon ECS)

Pour plus d'informations sur les fournisseurs de capacité d'instances gérées Amazon ECS, consultez la section Fournisseurs de capacité d'instances gérées Amazon ECS.

Ajout de balises à une instance de conteneur Amazon EC2 pour Amazon ECS

Vous pouvez associer des balises à vos instances de conteneur Amazon EC2 pour Amazon ECS en utilisant l'une des méthodes suivantes :

- Méthode 1 – Lorsque vous créez l'instance de conteneur avec l'API, la CLI ou la console Amazon EC2, spécifiez les balises en transmettant les données utilisateur à l'instance à l'aide du paramètre de configuration de l'agent de conteneur `ECS_CONTAINER_INSTANCE_TAGS`. Cela crée des balises qui sont associées uniquement à l'instance de conteneur dans Amazon ECS ; elles ne

peuvent pas être répertoriés à l'aide de l'API Amazon EC2. Pour de plus amples informations, veuillez consulter Démarrage des instances de conteneurs Amazon ECS Linux pour transmettre des données.

> **Important**
>
> Si vous lancez vos instances de conteneur à l'aide d'un groupe Amazon EC2 Auto Scaling, vous devez utiliser le paramètre de configuration de l'agent ECS_CONTAINER_INSTANCE_TAGS pour ajouter des balises. Cela est dû à la manière dont les balises sont ajoutées aux instances Amazon EC2 lancées à l'aide des groupes Auto Scaling.

Voici un exemple de script de données utilisateur qui associe des étiquettes avec votre instance de conteneur :

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
EOF
```

- Méthode 2 — Lorsque vous créez votre instance de conteneur à l'aide de l'API Amazon EC2, de la CLI ou de la console, spécifiez d'abord les balises à l'aide du paramètre `TagSpecification.N`. Transmettez ensuite les données utilisateur à l'instance à l'aide du paramètre de configuration de l'agent de conteneur `ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM`. Cela les propage depuis Amazon EC2 vers Amazon ECS.

  L'exemple suivant présente un script de données utilisateur qui propage les balises associées à une instance Amazon EC2 et enregistre l'instance avec un cluster nommé `MyCluster`.

  ```
  #!/bin/bash
  cat <<'EOF' >> /etc/ecs/ecs.config
  ECS_CLUSTER=MyCluster
  ECS_CONTAINER_INSTANCE_PROPAGATE_TAGS_FROM=ec2_instance
  EOF
  ```

  Pour donner l'autorisation aux balises de l'instance de conteneur de propager les données depuis Amazon EC2 vers Amazon ECS, ajoutez manuellement les autorisations suivantes comme

stratégie en ligne au rôle IAM de l'instance de conteneur Amazon ECS. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `ec2:DescribeTags`

L'exemple suivant de stratégie ajoute ces autorisations.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
             "ec2:DescribeTags"
            ],
            "Resource": "*"
        }
    ]
}
```

Ajout de balises à une instance de conteneur externe pour Amazon ECS

Vous pouvez associer des balises à vos instances de conteneur externes pour Amazon ECS en utilisant l'une des méthodes suivantes.

- Méthode 1 – Avant d'exécuter le script d'installation pour enregistrer votre instance externe auprès de votre cluster, créez ou modifiez le fichier de configuration de l'agent de conteneur Amazon ECS à l'adresse `/etc/ecs/ecs.config` et ajoutez le paramètre de configuration `ECS_CONTAINER_INSTANCE_TAGS` de l'agent de conteneur. Cela permet de créer des balises associées à l'instance externe.

  Voici un exemple de syntaxe.

```
ECS_CONTAINER_INSTANCE_TAGS={"tag_key": "tag_value"}
```

- Méthode 2 — Une fois que votre instance externe est enregistrée dans votre cluster, vous pouvez utiliser le Console de gestion AWS pour ajouter des balises. Pour de plus amples informations, veuillez consulter Ajout de balises aux ressources existantes (console Amazon ECS).

Rapports d'utilisation d'Amazon ECS

AWS fournit un outil de reporting appelé Cost Explorer que vous pouvez utiliser pour analyser le coût et l'utilisation de vos ressources Amazon ECS.

Vous pouvez utiliser Cost Explorer pour afficher les graphiques de votre utilisation et de vos coûts. Vous pouvez afficher les données des 13 derniers mois et prévoir vos dépenses pour les trois prochains mois. Vous pouvez utiliser Cost Explorer pour afficher des schémas de vos dépenses en ressources AWS au fil du temps. Par exemple, vous pouvez l'utiliser pour identifier les zones qui méritent d'être approfondies et connaître les tendances que vous pouvez utiliser pour comprendre vos coûts. Vous pouvez également spécifier des plages de temps pour les données et afficher des données temporelles par jour ou par mois.

Les données de mesure de votre rapport de coût et d'utilisation illustre l'utilisation parmi toutes vos tâches Amazon ECS. Les données de métrique incluent l'utilisation du processeur comme `vCPU-Hours` et l'utilisation de la mémoire comme `GB-Hours` pour chaque tâche exécutée. La manière dont ces données sont présentées dépend de l'option de calcul de la tâche.

Pour les tâches utilisant Fargate, la colonne `lineItem/Operation` affiche `FargateTask` et vous permet de connaître le coût associé à chaque tâche.

Pour les tâches utilisant EC2, la colonne `lineItem/Operation` affiche `ECSTask-EC2` et les tâches ne seront pas associées à un coût direct. Les données de mesure indiquées dans le rapport, telles que l'utilisation de la mémoire, représentent le total des ressources réservées par la tâche au cours de la période de facturation que vous spécifiez. Vous pouvez utiliser ces données pour déterminer le coût de votre cluster sous-jacent d'instances Amazon EC2. Les données de coût et d'utilisation de vos instances Amazon EC2 seront répertoriées séparément sous le service Amazon EC2.

Vous pouvez également utiliser les balises gérées par Amazon ECS pour identifier le service ou cluster auquel chaque tâche appartient. Pour de plus amples informations, veuillez consulter Mise à jour des balises pour la facturation.

> **Important**
>
> Les données de mesure sont uniquement visualisables pour les tâches lancées à compter du 16 novembre 2018. Les tâches lancées avant cette date n'affichent pas les données de mesure.

Voici un exemple des champs que vous pouvez trier par données d'attribution de coût dans Cost Explorer.

- Nom du cluster
- Nom du service
- Étiquettes de ressources
- Type de lancement
- Région AWS
- Type d'utilisation

Pour plus d'informations sur la création d'un rapport sur les AWS coûts et l'utilisation, consultez le rapport sur les AWS coûts et l'utilisation dans le guide de AWS Billing l'utilisateur.

Task-level Rapports sur les coûts et l'utilisation

AWS Cost Management peut fournir des données d'utilisation du processeur et de la mémoire AWS Cost and Usage Report pour chaque tâche sur Amazon ECS, y compris les tâches sur Fargate et les tâches sur EC2. Ces données sont appelées données de répartition des coûts fractionnés. Vous pouvez utiliser ces données pour analyser les coûts et l'utilisation des applications. En outre, vous pouvez fractionner et répartir les coûts entre les différentes unités commerciales et équipes à l'aide de balises de répartition des coûts et de catégories de coûts. Pour plus d'informations sur les données de répartition des coûts fractionnés, voir Comprendre les données de répartition des coûts partagés dans le Guide de AWS Cost and Usage Report l'utilisateur.

Vous pouvez activer les données de répartition des coûts fractionnés au niveau des tâches pour le compte dans l' AWS Cost Management Console. Si vous avez un compte de gestion (payeur), vous pouvez choisir d'appliquer cette configuration à chaque compte associé depuis le compte payeur.

Une fois que vous avez configuré les données de répartition des coûts, des colonnes supplémentaires apparaîtront sous l'LineItemen-tête fractionné dans le rapport. Pour plus

d'informations, voir les détails des articles divisés dans le guide de AWS Cost and Usage Report l'utilisateur

Pour les tâches sur EC2, ces données fractionnent le coût de l'instance EC2 en fonction de l'utilisation des ressources ou des réservations et des ressources restantes de l'instance.

Les conditions requises sont les suivantes :

- Définissez le paramètre de configuration de l'agent Amazon ECS `ECS_DISABLE_METRICS` sur `false`.

  Lorsque ce paramètre est défini`false`, l'agent Amazon ECS envoie des métriques à Amazon CloudWatch. Sous Linux, ce paramètre est défini `false` par défaut et les métriques sont envoyées à CloudWatch. Sous Windows, ce paramètre est défini `true` par défaut. Vous devez donc le modifier `false` CloudWatch pour envoyer les mesures AWS Cost Management à utiliser. Pour en savoir plus sur la configuration de l'agent ECS, veuillez consulter Configuration de l'agent de conteneur Amazon ECS.
- La version minimale de Docker pour des métriques fiables est Docker version 20.10.13 et ultérieure, incluse dans Amazon AMI ECS-optimized 20220607 et versions ultérieures.

Pour utiliser les données de répartition des coûts fractionnés, vous devez créer un rapport et sélectionner Données de répartition des coûts fractionnés. Pour plus d'informations, consultez la section Création de rapports sur les coûts et l'utilisation dans le guide de AWS Cost and Usage Report l'utilisateur.

AWS Cost Management calcule les données de répartition des coûts en fonction du processeur de la tâche et de l'utilisation de la mémoire. AWS Cost Management peut utiliser le processeur de tâche et la réservation de mémoire au lieu de l'utilisation, si l'utilisation n'est pas disponible. Si vous constatez que le CUR utilise les réservations, vérifiez que vos instances de conteneur répondent aux conditions requises et que les mesures d'utilisation des ressources des tâches apparaissent dans CloudWatch.

Surveillance d'Amazon ECS

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon ECS et de votre AWS solutions. Vous devez collecter des données de surveillance provenant de toutes les parties de votre AWS solution qui vous permet de déboguer plus facilement une panne multipoint si elle survient. Avant de commencer à surveiller Amazon ECS, créez un plan de surveillance qui comprend les réponses aux questions suivantes :

- Quels sont les objectifs de la surveillance ?
- Quelles sont les ressources à surveiller ?
- À quelle fréquence les ressources doivent-elles être surveillées ?
- Quels outils de surveillance utiliser ?
- Qui exécute les tâches de supervision ?
- Qui doit être informé en cas de problème ?

Les métriques disponibles dépendent de l’option de calcul des tâches et des services dans vos clusters. Si vous utilisez Fargate pour vos services, les métriques d’utilisation de l’UC et de la mémoire sont fournies pour faciliter leur surveillance. Pour EC2, vous devez surveiller les instances EC2 que vous possédez et qui constituent votre infrastructure sous-jacente. D’autres métriques d’utilisation et de réserve de l’UC et de la mémoire sont disponibles au niveau du cluster, du service et de la tâche.

L'étape suivante consiste à établir une référence de performances Amazon ECS normales dans votre environnement, en mesurant la performance à divers moments et dans diverses conditions de charge. Lorsque vous surveillez Amazon ECS, conservez les données d'historique de surveillance afin de pouvoir les comparer aux données de performances actuelles, d'identifier les modèles de performances normales et les anomalies de performances, et de concevoir des méthodes pour résoudre les problèmes.

Pour établir une référence, vous devez, au moins, superviser les éléments suivants :

- Métriques de réservation et d'utilisation de l'UC et de la mémoire pour les clusters Amazon ECS
- Métriques d'utilisation de l'UC et de la mémoire pour les services Amazon ECS

  Pour plus d’informations, consultez Affichage des métriques Amazon ECS.

Pratiques exemplaires en matière de surveillance Amazon ECS

Utilisez les pratiques exemplaires suivantes pour surveiller Amazon ECS.

- Faites de la surveillance une priorité pour résoudre les petits problèmes avant qu'ils n'empirent
- Élaborez un plan de surveillance qui inclut les réponses aux questions suivantes
  - Quels sont les objectifs de la surveillance ?
  - Quelles sont les ressources à surveiller ?
  - À quelle fréquence les ressources doivent-elles être surveillées ?
  - Quels outils de surveillance utiliser ?
  - Qui exécute les tâches de supervision ?
  - Qui doit être informé en cas de problème ?
- Automatisez la surveillance autant que possible.
- Consultez les fichiers journaux Amazon ECS. Pour plus d'informations, consultez Affichage du journal d'agent de conteneur Amazon ECS.
- Utilisez Runtime Monitoring pour protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. Pour plus d'informations, consultez Identification des comportements non autorisés grâce à la surveillance d'exécution.

Outils de surveillance pour Amazon ECS

AWS fournit différents outils que vous pouvez utiliser pour surveiller Amazon ECS. Vous pouvez configurer certains outils pour qu'ils effectuent la supervision automatiquement, tandis que d'autres nécessitent une intervention manuelle. Nous vous recommandons d'automatiser le plus possible les tâches de supervision.

Outils de surveillance automatique

Vous pouvez utiliser les outils de surveillance automatique pour surveiller Amazon ECS et signaler un problème éventuel :

- Amazon CloudWatch alarmes : surveillez une seule métrique pendant une période que vous spécifiez et effectuez une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de périodes. L'action est une notification envoyée à une rubrique Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2

Auto Scaling. CloudWatch les alarmes ne déclenchent pas d'actions simplement parce qu'elles se trouvent dans un état particulier ; cet état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour plus d'informations, consultez Surveillez Amazon ECS à l'aide CloudWatch.

Pour les services dont les tâches utilisent Fargate, vous pouvez utiliser CloudWatch des alarmes permettant d'adapter et de réduire les tâches de votre service en fonction de CloudWatch des mesures, telles que l'utilisation du processeur et de la mémoire. Pour plus d'informations, consultez Mise à l'échelle automatique de votre service Amazon ECS.

Pour les clusters dont les tâches ou les services utilisent EC2, vous pouvez utiliser CloudWatch alarmes pour intégrer et redimensionner les instances de conteneurs en fonction de CloudWatch des mesures, telles que la réservation de la mémoire du cluster.

Pour vos instances de conteneurs qui ont été lancées avec Amazon ECS-optimized Amazon Linux AMI, vous pouvez utiliser CloudWatch Journaux pour consulter les différents journaux de vos instances de conteneurs en un seul endroit pratique. Vous devez installer le CloudWatch agent sur vos instances de conteneurs. Pour plus d'informations, voir Téléchargez et configurez le CloudWatch agent à l'aide de la ligne de commande dans le Amazon CloudWatch Guide de l'utilisateur. Vous devez également ajouter la stratégie `ECS-CloudWatchLogs` au rôle `ecsInstanceRole`. Pour plus d'informations, consultez Autorisations pour la surveillance des instances de conteneur.

- Amazon CloudWatch Journaux : surveillez, stockez et accédez aux fichiers journaux depuis les conteneurs dans le cadre de vos tâches Amazon ECS en spécifiant `awslogs` enregistrez le pilote dans vos définitions de tâches. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à CloudWatch.

  Vous pouvez également surveiller, stocker et accéder au système d'exploitation et aux fichiers journaux de l'agent de conteneur Amazon ECS à partir de vos instances de conteneur Amazon ECS. Cette méthode d'accès aux journaux peut être utilisée pour les conteneurs utilisant EC2.

- Amazon CloudWatch Événements : associez les événements et dirigez-les vers une ou plusieurs fonctions ou flux cibles pour apporter des modifications, capturer des informations d'état et prendre des mesures correctives. Pour plus d'informations, voir Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge dans ce guide et EventBridge est l'évolution d'Amazon CloudWatch Évènements dans le Amazon EventBridge Guide de l'utilisateur.

- Container Insights : collectez, agrégez et résumez les métriques et les journaux de vos applications conteneurisées et microservices. Container Insights collecte des données en tant qu'événements du journal des performances utilisant le format de métrique intégrée. Ces événements du journal des performances sont des entrées qui utilisent un schéma JSON structuré qui permet aux données à haute cardinalité d'être intégrées et stockées à grande échelle. À partir de ces données, CloudWatch crée des mesures agrégées au niveau du cluster, de la tâche et du service comme CloudWatch métriques. Les statistiques collectées par Container Insights sont disponibles dans CloudWatch tableaux de bord automatiques, et sont également consultables dans la section Métriques du CloudWatch console.
- AWS CloudTrail surveillance des journaux — Partagez des fichiers journaux entre les comptes, surveillez CloudTrail enregistrer les fichiers en temps réel en les envoyant à CloudWatch Journalise, écrit des applications de traitement des journaux en Java et vérifie que vos fichiers journaux n'ont pas changé après leur livraison en CloudTrail. Pour plus d'informations, voir Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail dans ce guide, et Travailler avec CloudTrail Fichiers journaux dans le AWS CloudTrail Guide de l'utilisateur.
- Surveillance d'exécution : détectez les menaces pesant sur les clusters et les conteneurs au sein de votre environnement AWS . Runtime Monitoring utilise un GuardDuty agent de sécurité qui ajoute de la visibilité à l'exécution des charges de travail Amazon ECS individuelles, par exemple, l'accès aux fichiers, l'exécution des processus et les connexions réseau.

Outils de surveillance manuelle

Une autre partie importante de la surveillance d'Amazon ECS consiste à surveiller manuellement les articles que CloudWatch les alarmes ne couvrent pas. Le CloudWatch, Trusted Advisor, et autres AWS les tableaux de bord de console fournissent une vue d'ensemble de l'état de votre AWS environnement. Nous vous recommandons de vérifier également les fichiers journaux de vos instances de conteneur et les conteneurs de vos tâches.

- Console Amazon ECS :
  - Métriques de cluster pour EC2
  - Métriques de service
  - Statut d'intégrité du service
  - Événements de déploiements de service
- CloudWatch page d'accueil :
  - Alarmes et statuts en cours

  - Graphiques des alarmes et des ressources
  - Statut d'intégrité du service

  De plus, vous pouvez utiliser CloudWatch pour effectuer les opérations suivantes :

  - Créer des tableaux de bord personnalisés pour surveiller les services de votre choix.
  - Données de métriques de graphiques pour résoudre les problèmes et découvrir les tendances.
  - Recherchez et parcourez tous vos AWS mesures relatives aux ressources.
  - Créer et Modifier des alarmes pour être informé des problèmes.
- Surveillance de l'état du conteneur : il s'agit de commandes qui s'exécutent localement sur un conteneur et valident l'état et la disponibilité des applications. Vous les configurez par conteneur dans votre définition de tâche.
- AWS Trusted Advisor peut vous aider à surveiller votre AWS des ressources pour améliorer les performances, la fiabilité, la sécurité et la rentabilité. Quatre Trusted Advisor les chèques sont disponibles pour tous les utilisateurs ; plus de 50 chèques sont disponibles pour les utilisateurs disposant d'un plan de support Business ou Enterprise. Pour plus d'informations, consultez AWS Trusted Advisor.

  Trusted Advisor possède les vérifications suivantes relatives à Amazon ECS :

  - Une tolérance aux pannes qui indique qu'un service est exécuté dans une seule zone de disponibilité.
  - Une tolérance aux pannes qui indique que vous n'avez pas utilisé la stratégie de placement réparti sur plusieurs zones de disponibilité.
- Optimiseur de calcul AWS est un service qui analyse les paramètres de configuration et d'utilisation de votre AWS ressources. Il indique si vos ressources sont optimales et génère des recommandations d'optimisation afin de réduire les coûts et d'améliorer les performances de vos charges de travail.

  Pour plus d'informations, consultez Optimiseur de calcul AWS recommandations pour Amazon ECS.
- Serveur Amazon ECS MCP : utilisez des assistants IA pour surveiller les déploiements, résoudre les problèmes liés aux conteneurs et inspecter les configurations à l'aide du langage naturel. Le serveur MCP fournit une visibilité en temps réel sur l'état du service, analyse les échecs de tâches et accélère la résolution des problèmes. Pour plus d'informations, consultez Serveur Amazon ECS MCP.

Surveillez Amazon ECS à l'aide CloudWatch

Vous pouvez surveiller vos ressources Amazon ECS à l'aide d'Amazon CloudWatch, qui collecte et traite les données brutes d'Amazon ECS pour les transformer en métriques lisibles en temps quasi réel. Ces statistiques sont enregistrées pour une durée de deux semaines. Vous pouvez, par conséquent, accéder aux informations historiques et mieux comprendre la façon dont vos clusters ou vos services fonctionnent. Les données métriques Amazon ECS sont automatiquement envoyées à CloudWatch par périodes d'une minute. Pour plus d'informations sur CloudWatch, consultez le Amazon CloudWatch Guide de l'utilisateur.

Amazon ECS fournit gratuitement des métriques pour les clusters et les services. Moyennant un coût supplémentaire, vous pouvez activer Amazon ECS CloudWatch Container Insights pour votre cluster pour les métriques par tâche, notamment l'utilisation du processeur, de la mémoire et du système de fichiers EBS. Pour plus d'informations sur Container Insights, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation d'Amazon ECS CloudWatch métriques.

- Tout service Amazon ECS hébergé sur Fargate possède CloudWatch Métriques d'utilisation du processeur et de la mémoire automatiquement, vous n'avez donc pas besoin de prendre de mesures manuelles.

- Pour toute tâche ou service Amazon ECS hébergé sur des instances Amazon EC2, l'instance Amazon EC2 nécessite une version `1.4.0` ou version ultérieure (Linux) ou `1.0.0` ou version ultérieure (Windows) de l'agent de conteneur pour CloudWatch métriques à générer. Nous vous recommandons toutefois d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS.
- La version minimale de Docker pour une fiabilité CloudWatch metrics est la version de Docker `20.10.13` et plus récents.
- Vos instances Amazon EC2 nécessitent également l'autorisation `ecs:StartTelemetrySession` pour le rôle IAM avec lequel vous les lancez. Si vous avez créé votre rôle IAM d'instance de conteneur Amazon ECS auparavant CloudWatch des mesures étaient disponibles pour Amazon ECS, il se peut que vous deviez ajouter cette autorisation. Pour plus d'informations sur le rôle

IAM de l'instance de conteneur et l'association de la politique IAM gérée pour les instances de conteneur, consultez la section Rôle IAM d'instance de conteneur Amazon ECS.

- Vous pouvez désactiver CloudWatch collecte de métriques sur vos instances Amazon EC2 en paramétrant `ECS_DISABLE_METRICS=true` dans votre configuration d'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Métriques recommandées

Amazon ECS fournit gratuitement CloudWatch des indicateurs que vous pouvez utiliser pour surveiller vos ressources. Ces métriques permettent de mesurer la réservation d'UC et de mémoire, l'utilisation de l'UC, de la mémoire et du système de fichiers EBS dans l'ensemble de votre cluster, ainsi que l'utilisation de l'UC, de la mémoire et du système de fichiers EBS sur les services de vos clusters. Pour vos charges de travail GPU, vous pouvez mesurer votre réservation GPU sur votre cluster.

L'infrastructure sur laquelle vos tâches Amazon ECS sont hébergées dans vos clusters détermine les métriques disponibles. Pour les tâches hébergées sur l'infrastructure Fargate, Amazon ECS fournit des métriques d'utilisation de l'UC, de la mémoire et du système de fichiers EBS pour faciliter la surveillance de vos services. Pour les tâches hébergées sur des instances EC2, Amazon ECS fournit des métriques de réserves d'UC, de mémoire et de GPU, ainsi que des métriques d'utilisation de l'UC et de la mémoire au niveau du cluster et du service. Vous devez surveiller séparément les instances Amazon EC2 qui constituent votre infrastructure sous-jacente. Pour plus d'informations sur la surveillance de vos instances EC2, consultez la section Surveillance Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Pour plus d'informations sur les alarmes recommandées à utiliser avec Amazon ECS, consultez l'une des rubriques suivantes dans Amazon CloudWatch Guide de l'utilisateur des journaux:

- Amazon ECS
- Amazon ECS avec Container Insights

Affichage des métriques Amazon ECS

Une fois que les ressources sont en cours d'exécution dans votre cluster, vous pouvez consulter les métriques sur Amazon ECS et CloudWatch consoles. La console Amazon ECS fournit une vue maximum, minimum et moyenne de 24 heures des métriques de votre cluster et du service. La

console CloudWatch fournit un affichage personnalisable et précis de vos ressources, ainsi que le nombre de tâches en cours d'exécution dans un service.

Console Amazon ECS

Les métriques d'utilisation des unités UC et de la mémoire Amazon ECS service sont disponibles sur la console Amazon ECS. La vue des métriques de service présente les valeurs moyennes, minimales et maximales pour la période de 24 heures qui précède, avec des points de données disponibles à intervalle de 5 minutes. Pour plus d'informations, consultez Métriques d'utilisation d'un service Amazon ECS.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sélectionnez le cluster dans lequel vous voulez afficher les métriques.
3. Déterminez les métriques à afficher.

Pour afficher les métriques de	Étapes
Clusters	Sur la page des détails du cluster, choisissez l'onglet Métriques. Un lien est également fourni vers le CloudWatch console pour visualiser votre CloudWatch Statistiques de Container Insights si elles sont activées.
Services	Sur la page des détails du cluster, sous l'onglet Services, sélectionnez le service. Les métriques sont alors disponibles sous l'onglet État et métriques.

CloudWatch console

Pour Fargate, les statistiques du service Amazon ECS peuvent également être consultées sur CloudWatch console. La console fournit la vue la plus détaillée des métriques Amazon ECS et vous pouvez personnaliser les vues en fonction de vos besoins. Vous pouvez consulter l'utilisation du service et le nombre de tâches en cours d'exécution du service.

Pour les fournisseurs de capacité EC2, les métriques du cluster et des services Amazon ECS peuvent également être consultées sur le CloudWatch console. La console fournit la vue la plus détaillée des métriques Amazon ECS et vous pouvez personnaliser les vues en fonction de vos besoins.

Pour plus d'informations sur la façon d'afficher les statistiques, consultez Afficher les statistiques disponibles le Amazon CloudWatch Guide de l'utilisateur.

Amazon ECS CloudWatch indicateurs

Vous pouvez utiliser CloudWatch des statistiques d'utilisation pour fournir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces statistiques pour visualiser votre utilisation actuelle des services sur CloudWatch graphiques et tableaux de bord.

Amazon ECS envoie des métriques à CloudWatch à intervalles d'une minute. Ces métriques sont collectées pour les ressources dont les tâches sont dans l'état `RUNNING`. Si aucune tâche n'est en cours d'exécution sur un cluster, un service ou une autre ressource, aucune métrique ne sera communiquée pour cette ressource pendant cette période. Par exemple, si vous avez un cluster avec un service mais que ce service ne comporte aucune tâche dans `RUNNING` état, aucune métrique ne sera envoyée à CloudWatch. De même, si vous disposez de deux services et que l'un d'eux exécute des tâches tandis que l'autre n'en exécute pas, seules les métriques du service exécutant des tâches seront envoyées.

Métrique	Description	Dimension valide	Statistiques utiles	Unit
CPUReservation	Pourcentage d'unités d'UC réservées dans le cluster ou le service.	`ClusterName` .	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	La réserve d'UC (filtrée par `ClusterName`) est calculée en divisant le nombre total d'unités d'UC réservées par les tâches Amazon ECS sur le cluster par le nombre total d'unités d'UC pour toutes les instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING` auront une incidence sur métriques de réserve d'UC. Cette méthode n'est prise en charge que pour les tâches hébergées sur			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	des instances Amazon EC2.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`CPUUtilization`	Pourcentage d'unités de processeur utilisées par le cluster, le service ou le démon géré. L'utilisation de l'UC au niveau du cluster (filtrées par `ClusterName`) est mesurée sur la base du nombre total d'unités d'UC en cours d'utilisation par les tâches Amazon ECS sur le cluster, divisé par le nombre total d'unités d'UC pour toutes les instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING`	`ClusterName`, `ServiceName`	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	auront une incidence sur métriques de réserve d'UC. La métrique au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une instance Amazon EC2. L'utilisation de l'UC au niveau du service (filtrée par `ClusterName` , `ServiceName`) est mesurée comme le nombre total d'unités d'UC utilisées par les tâches appartenant au service, divisé par le nombre total d'unités d'UC qui sont en réserve pour les tâches appartenant au service.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	La métrique de niveau de service est prise en charge pour les tâches hébergées sur des instances Amazon EC2 et Fargate. L'utilisation du processeur au niveau du démon (filtrée par `ClusterName` , `ServiceName`) est mesuré comme le nombre total d'unités de processeur utilisées par les tâches appartenant au démon géré, divisé par le nombre total d'unités de processeur réservées aux tâches appartenant au démon géré.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`MemoryReservation`	Pourcentage de mémoire qui est réservé en exécutant des tâches dans le cluster. La réserve de mémoire du cluster est mesurée comme la mémoire totale réservée par les tâches Amazon ECS sur le cluster, divisée par la quantité totale de mémoire pour toutes les instances Amazon EC2 enregistrées dans le cluster. Cette métrique ne peut être filtrée que par `ClusterName` . Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING`	`ClusterName` .	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	auront une incidence sur métriques de réserve de mémoire. La métrique de réserve de mémoire au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une instance Amazon EC2. **Note** Lors du calcul de l'utilisation de la mémoire, si elle `MemoryReservation` est spécifiée, elle est utilisée dans le calcul au lieu de la			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	mémoire totale.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`MemoryUtilization`	Pourcentage de mémoire utilisée par le cluster, le service ou le démon géré. L'utilisation de la mémoire au niveau du cluster (filtrée par `ClusterName`) est mesurée comme la mémoire totale utilisée par les tâches Amazon ECS sur le cluster, divisée par la mémoire totale de toutes les instances Amazon EC2 enregistrées dans le cluster. Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING` auront une incidence sur métriques d'utilisation de	`ClusterName` , `ServiceName`	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	mémoire. La métrique au niveau du cluster n'est prise en charge que pour les tâches hébergées sur une instance Amazon EC2. L'utilisation de la mémoire au niveau du service (filtrée par `ClusterName` , `ServiceName`) est mesurée comme la mémoire totale utilisée par les tâches appartenant au service, divisée par la mémoire totale réservée aux tâches appartenant au service. La métrique de niveau de service est prise en charge pour les tâches			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	hébergées sur des instances Amazon EC2 et Fargate. L'utilisation de la mémoire au niveau du démon (filtrée par `ClusterName` , `ServiceName`) est mesurée comme la mémoire totale utilisée par les tâches appartenant au démon géré, divisée par la mémoire totale réservée aux tâches appartenant au démon géré.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`EBSFilesystemUtilization`	Pourcentage du système de fichiers Amazon EBS utilisé par les tâches d'un service. La métrique d'utilisation du système de fichiers EBS au niveau du service (filtrée par `ClusterName` , `ServiceName`) est mesurée comme la quantité totale du système de fichiers EBS utilisée par les tâches appartenant au service, divisée par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches appartenant au service. La	`ClusterName` , `ServiceName`	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	métrique d'utilisation du système de fichiers EBS au niveau du service n'est disponible que pour les tâches hébergées sur des instances Amazon EC2 (utilisant la version de l'agent conteneur `1.79.0`) et Fargate (utilisant la version de plateforme `1.4.0`) auxquelles un volume EBS est associé. **Note** Pour les tâches hébergées sur Fargate, il existe un espace sur le disque			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	qui n'est utilisé que par Fargate. Il n'y a aucun coût associé à l'espace utilisé par Fargate, mais vous pourrez voir ce stockage supplémentaire à l'aide d'outils tels que `df`.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`GPUReservation`	Le pourcentage du total de GPU disponibles qui sont réservés en exécutant des tâches dans le cluster. La métrique de réserve de GPU au niveau du cluster est calculée en divisant le nombre de GPU réservés par les tâches Amazon ECS sur le cluster par le nombre total de GPU disponibles sur toutes les instances Amazon EC2 avec des GPU enregistrés dans le cluster. Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING` auront une incidence sur	`ClusterName`	Moyenne, Minimum, Maximum	Pourcentage

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	métriques de réserve de GPU.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ActiveConnectionCount`	Nombre total de connexions simultanées actives entre les clients et les proxys Amazon ECS Service Connect qui s'exécutent dans le cadre de tâches qui partagent le `DiscoveryName` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect. Dimensions valides : `DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`.	`DiscoveryName` et `DiscoveryName, ServiceName, ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`NewConnectionCount`	Nombre total de nouvelles connexions établies entre les clients et les proxys Service Connect Amazon ECS qui s'exécutent dans le cadre de tâches qui partagent le `DiscoveryName` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ProcessedBytes`	Nombre total d'octets de trafic entrant traités par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Octets

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`RequestCount`	Nombre de demandes de trafic entrant traitées par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect. Vous devez également configurer `appProtocol` dans le mappage des ports dans votre définition de tâche.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`GrpcRequestCount`	Nombre de demandes de trafic entrant gRPC traitées par les proxys Service Connect. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est `GRPC` dans le mappage des ports de la définition de la tâche.	`DiscoveryName` et `DiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_2XX_Count`	Nombre de codes de réponse HTTP numérotés de 200 à 299 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est HTTP ou HTTP2 dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche. Dimensions valides :			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_3XX_Count`	Nombre de codes de réponse HTTP numérotés de 300 à 399 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est HTTP ou HTTP2 dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_4XX_Count`	Nombre de codes de réponse HTTP numérotés de 400 à 499 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est HTTP ou HTTP2 dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`HTTPCode_Target_5XX_Count`	Nombre de codes de réponse HTTP numérotés de 500 à 599 générés par les applications dans le cadre de ces tâches. Ces tâches sont les cibles. Cette métrique ne compte que les réponses envoyées aux proxys Service Connect par les applications dans le cadre de ces tâches, et non les réponses envoyées directement. Cette métrique n'est disponible que si vous avez configuré Amazon ECS Service Connect et que le `appProtocol` est HTTP ou HTTP2 dans	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	le mappage des ports de la définition de la tâche.			
`RequestCountPerTarget`	Nombre moyen de demandes reçues par chaque cible qui partage le `DiscoveryName` sélectionné. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne	Nombre
`TargetProcessedBytes`	Nombre total d'octets traités par les proxys Service Connect. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum, Somme	Octets

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`TargetResponseTime`	Latence du traitement des demandes d'application. Temps écoulé, en millisecondes, entre le moment où la demande arrive au proxy Service Connect dans la tâche cible et le moment où une réponse de l'application cible arrive au proxy. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`TargetDiscoveryName` et `TargetDiscoveryName`, `ServiceName`, `ClusterName`	Moyenne, Minimum, Maximum	Millisecondes

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`ClientTLSNegotiationErrorCount`	Nombre total de fois où la connexion TLS a échoué. Cette métrique n'est utilisée que lorsque le protocole TLS est activé. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`DiscoveryName` et `DiscoveryName` , `ServiceName` , `ClusterName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`TargetTLSNegotiationErrorCount`	Le nombre total de fois où la connexion TLS a échoué en raison de certificats clients manquants, a échoué CA privée AWS vérifications, ou échec des vérifications SAN. Cette métrique n’est utilisée que lorsque le protocole TLS est activé. Cette métrique n'est disponible que si vous disposez d'Amazon ECS Service Connect.	`ServiceName` , `ClusterName` , `TargetDiscoveryName` et `TargetDiscoveryName`	Moyenne, Minimum, Maximum, Somme	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
`LiveTaskCount`	Le nombre de tâches actuellement en cours `ACTIVATING` , `RUNNING`, et `DEACTIVATING` états du cycle de vie du service. Tâches dans `ACTIVATING` les États se préparent à diffuser du trafic (par exemple, enregistrement auprès des groupes cibles de l'équilibreur de charge et découverte de services). Tâches dans `RUNNING` les états sont en cours d'exécution et disponibles pour recevoir du trafic et gérer les demandes. Tâches dans `DEACTIVATING` les États	`ClusterName` , `ServiceName`	Moyenne, Minimum, Maximum	Nombre

Métrique	Description	Dimension valide	Statistiques utiles	Unit
	épuisent les connexions et se désinscrivent des ressource s servant au trafic avant d'être arrêtés.			

Dimensions pour les métriques Amazon ECS

Les métriques Amazon ECS utilisent l'espace de noms `AWS/ECS` et fournissent des métriques pour les dimensions suivantes. Amazon ECS envoie uniquement des métriques pour les ressources dont les tâches sont à l'état `RUNNING`. Par exemple, si vous avez un cluster avec un service mais que ce service n'a aucune tâche avec l'état `RUNNING`, aucune métrique n'est envoyée à CloudWatch. Si vous disposez de deux services et que l'un d'eux seulement a des tâches en cours d'exécution, seules les métriques du service avec des tâches en cours d'exécution seront envoyées.

Dimension	Définition
`ClusterName`	Cette dimension filtre les données que vous demandez pour toutes les ressources dans un cluster donné. Toutes les métriques Amazon ECS sont filtrées par `ClusterName` .
`ServiceName`	Cette dimension filtre les données que vous demandez pour toutes les ressources d'un service ou d'un démon spécifique au sein d'un cluster spécifié. Pour les métriques des démons, `ServiceName` la valeur de dimension a le préfixe `daemon:` suivi du nom du démon. Par exemple, un démon nommé `my-daemon` aurait un `ServiceName` valeur dimensionnelle de `daemon:my-daemon` .

Dimension	Définition
`DiscoveryName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié dans tous les clusters Amazon ECS. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.
`DiscoveryName, ServiceName, ClusterName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié pour les tâches portant ce nom de découverte et créées par ce service dans ce cluster. Utilisez cette dimension pour voir les métriques du trafic entrant d'un service spécifique, si vous avez réutilisé le même nom de découverte dans plusieurs services dans des espaces de noms différents. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

Dimension	Définition
`TargetDiscoveryName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié dans tous les clusters Amazon ECS. À la différence de `DiscoveryName` , ces métriques de trafic mesurent uniquement le trafic entrant vers ce `DiscoveryName` issu d'autres tâches Amazon ECS ayant une configuration Service Connect dans cet espace de noms. Cela inclut les tâches effectuées par les services avec une configuration Service Connect client uniquement ou client-serveur. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

Dimension	Définition
`TargetDiscoveryName, ServiceName, ClusterName`	Cette dimension filtre les données que vous demandez pour les métriques de trafic vers un nom de découverte Service Connect spécifié, mais ne compte que le trafic des tâches créées par ce service dans ce cluster. Utilisez cette dimension pour voir les métriques de trafic entrant provenant d'un client spécifique dans un autre service. À la différence de `DiscoveryName, ServiceName, ClusterName` , ces métriques de trafic mesurent uniquement le trafic entrant vers ce `DiscoveryName` issu d'autres tâches Amazon ECS ayant une configuration Service Connect dans cet espace de noms. Cela inclut les tâches effectuées par les services avec une configuration Service Connect client uniquement ou client-serveur. Notez qu'un port spécifique d'un conteneur en cours d'exécution peut avoir plusieurs noms de découverte.

AWS Fargate métriques d’utilisation

Vous pouvez utiliser CloudWatch des statistiques d'utilisation pour fournir une visibilité sur l'utilisation des ressources de votre compte. Utilisez ces statistiques pour visualiser votre utilisation actuelle des services sur CloudWatch graphiques et tableaux de bord.

AWS Fargate les mesures d'utilisation correspondent à AWS quotas de service. Vous pouvez configurer des alarmes qui vous alertent lorsque votre utilisation approche d’un quota de service. Pour plus d’informations sur les quotas de service Fargate, consultez la section Points de terminaison et quotas Amazon ECS dans la Référence générale d'Amazon Web Services.

AWS Fargate publie les mesures suivantes dans le `AWS/Usage` espace de noms.

Métrique	Description
`ResourceCount`	Nombre total des ressources spécifiées exécutées sur votre compte. La ressource est définie par les dimensions associées à la métrique.

Les dimensions suivantes permettent d'affiner les métriques d'utilisation publiées par AWS Fargate.

Dimension	Description
`Service`	Le nom du AWS service contenant la ressource. Pour les métriques d'utilisation d' AWS Fargate , la valeur de cette dimension est `Fargate`.
`Type`	Type d’entité faisant l’objet d’un rapport. Actuellement, la seule valeur valide pour AWS Fargate les mesures d'utilisation sont `Resource`.
`Resource`	Type de ressource en cours d’exécution. Type de ressource en cours d’exécution. Actuellement, la seule valeur valide pour AWS Fargate les mesures d'utilisation sont `vCPU` qui renvoie des informations sur les instances en cours d'exécution.
`Class`	Classe de ressource suivie. Classe de ressource suivie. Pour AWS Fargate métriques d'utilisation avec vCPU comme valeur de la dimension Resource, les valeurs valides sont `Standard/OnDemand` et `Standard/Spot` .

Vous pouvez utiliser la console Service Quotas pour visualiser votre utilisation sur un graphique et configurer des alarmes qui vous alertent lorsque AWS Fargate l'utilisation approche un quota de service. Pour plus d'informations sur la création d'un CloudWatch alarme pour vous avertir lorsque vous êtes proche d'un seuil de valeur de quota, voir Quotas de service et Amazon CloudWatch alarmes dans le Service Quotas Guide de l'utilisateur

.

Métriques de réserve de cluster Amazon ECS

Les métriques de réservation de cluster correspondent au pourcentage d'UC, de mémoire et de GPU réservé par toutes les tâches Amazon ECS sur un cluster par rapport aux ressources d'UC, de mémoire et de GPU regroupées qui ont été enregistrées pour chaque instance de conteneur active dans le cluster. Seules les instances de conteneur ayant le statut `ACTIVE` ou `DRAINING` affectent les métriques de réservation de cluster. Cette métrique n'est utilisée que sur les clusters dont les tâches ou les services sont hébergés sur des instances EC2. Cette fonctionnalité n'est pas prise en charge sur les clusters dont les tâches sont hébergées sur AWS Fargate.

```
                              (Total CPU units reserved by tasks in cluster) x 100
Cluster CPU reservation =
 ---------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                 (Total MiB of memory reserved by tasks in cluster x
 100)
Cluster memory reservation =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

```
                                 (Total GPUs reserved by tasks in cluster x 100)
Cluster GPU reservation =
 ------------------------------------------------------------------
                              (Total GPUs registered by container instances in cluster)
```

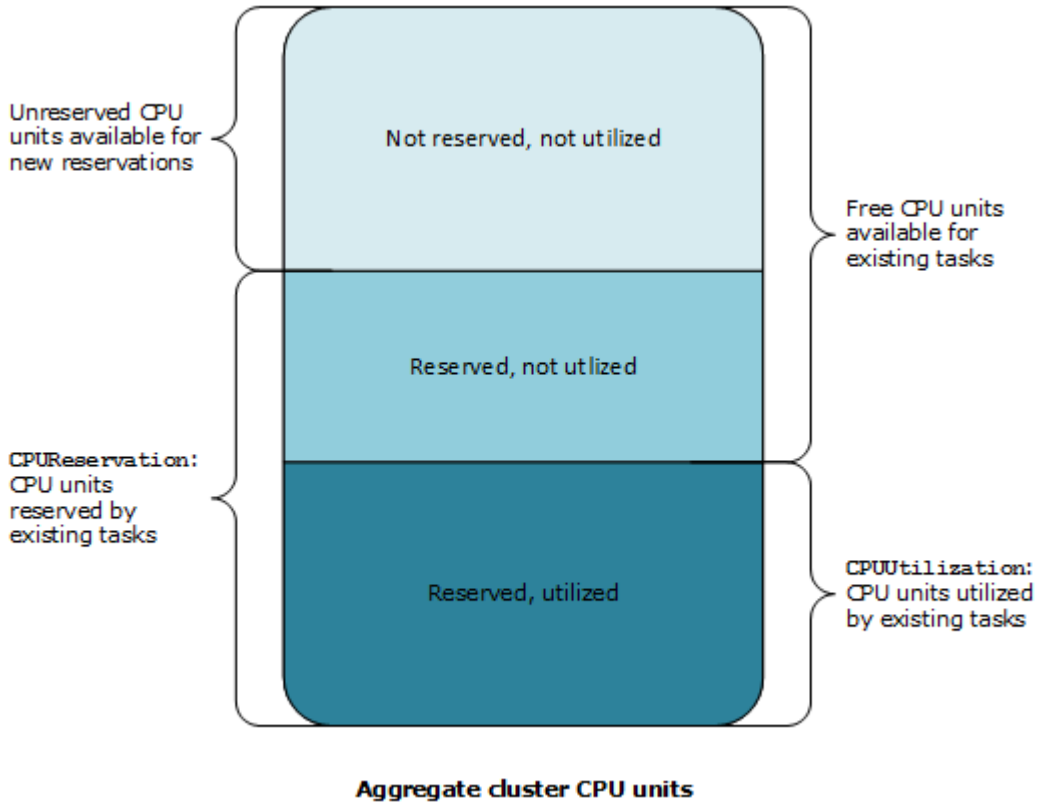
Lorsque vous exécutez une tâche dans un cluster, Amazon ECS analyse la définition de la tâche et réserve les unités d'UC, les Mio de mémoire et les GPU regroupés qui sont spécifiés dans ses définitions de conteneur. Chaque minute, Amazon ECS calcule le nombre d'unités d'UC, les Mio de mémoire et les GPU qui sont actuellement réservés pour chaque tâche en cours d'exécution dans le cluster. La quantité totale de CPU, de mémoire et de GPU réservée à toutes les tâches exécutées sur le cluster est calculée et ces chiffres sont transmis à CloudWatch en pourcentage du total des ressources enregistrées pour le cluster. Si vous spécifiez une limite flexible (`memoryReservation`) dans la définition de tâche, elle est utilisée pour calculer la quantité de mémoire réservée. Sinon, la limite stricte (`memory`) est utilisée. Le total des Mio de mémoire réservés par les tâches d'un cluster inclut également la taille du volume du système de fichiers temporaire

(`tmpfs`), et `sharedMemorySize`, si elle est définie dans la définition de la tâche. Pour en savoir plus sur les limites strictes et flexibles, la taille de mémoire partagée et la taille de volume tmpfs, veuillez consulter Paramètres de définition de tâche (langue française non garantie).

Prenons l'exemple d'un cluster qui possède deux instances de conteneur actives enregistrées : une instance `c4.4xlarge` et une instance `c4.large`. L'instance `c4.4xlarge` s'enregistre dans le cluster avec 16 384 unités d'UC et 30 158 Mio de mémoire. L'instance `c4.large` s'enregistre avec 2 048 unités d'UC et 3 768 Mio de mémoire. Les ressources globales de ce cluster sont de 18 432 unités d'UC et de 33 926 Mio de mémoire.

Si une définition de tâche se réserve 1 024 unités d'UC et 2 048 Mio de mémoire, et que dix tâches sont lancées avec cette définition de tâche sur ce cluster (et qu'aucune autre tâche n'est en cours d'exécution), un total de 10 240 unités d'UC et 20 480 Mio de mémoire est réservé. Ceci est rapporté à CloudWatch sous la forme d'une réservation de 55 % du processeur et de 60 % de la mémoire réservée pour le cluster.

L'illustration suivante présente le nombre total d'unités d'UC enregistrées dans un cluster et la signification de leur réservation et de leur utilisation pour les tâches existantes et le placement des nouvelles tâches. Les blocs inférieur (Réservé, utilisé) et central (Réservé, non utilisé) représentent le nombre total d'unités de processeur réservées aux tâches existantes exécutées sur le cluster, ou `CPUReservation` CloudWatch métrique. Le bloc inférieur représente les unités de processeur réservées que les tâches en cours d'exécution utilisent réellement sur le cluster, ou `CPUUtilization` CloudWatch métrique. Le bloc supérieur représente les unités d'UC qui ne sont pas réservées par des tâches existantes. Ces unités d'UC sont disponibles pour le placement de nouvelles tâches. Les tâches existantes peuvent également utiliser ces unités d'UC non réservées, si leur besoin en ressources processeur augmente. Pour plus d'informations, consultez la documentation relative aux paramètres de définition de tâche `cpu`.



Métriques d'utilisation de cluster Amazon ECS

Les métriques d'utilisation de cluster sont disponibles pour l'UC, la mémoire et, lorsqu'un volume EBS est associé à vos tâches, l'utilisation du système de fichiers EBS. Ces métriques ne sont disponibles que pour les clusters dont les tâches ou les services sont hébergés sur des instances Amazon EC2. Ils ne sont pas pris en charge sur les clusters dont les tâches sont hébergées sur AWS Fargate.

Métrique d'utilisation de l'UC et de la mémoire au niveau du cluster Amazon ECS

L'utilisation de l'UC et de la mémoire est mesurée en pourcentage de l'UC et de la mémoire utilisés par toutes les tâches d'un cluster par rapport à l'UC et à la mémoire globaux enregistrés pour chaque instance Amazon EC2 active enregistrée dans le cluster. Seules les instances Amazon EC2 dont le statut est `ACTIVE` ou `DRAINING` auront une incidence sur les métriques d'utilisation du cluster.

```
                                (Total CPU units used by tasks in cluster) x 100
Cluster CPU utilization =
 ------------------------------------------------------------
                           (Total CPU units registered by container instances in
 cluster)
```

```
                                    (Total MiB of memory used by tasks in cluster x
 100)
Cluster memory utilization =
 ------------------------------------------------------------------
                              (Total MiB of memory registered by container instances in
 cluster)
```

Chaque minute, l'agent de conteneur Amazon ECS sur chaque instance Amazon EC2 calcule le nombre d'unités d'UC et de Mio de mémoire actuellement utilisés pour chaque tâche en cours d'exécution sur cette instance Amazon EC2, et ces informations sont renvoyées à Amazon ECS. La quantité totale de processeur et de mémoire utilisée pour toutes les tâches exécutées sur le cluster est calculée et ces chiffres sont transmis à CloudWatch en pourcentage du total des ressources enregistrées pour le cluster.

Prenons l'exemple d'un cluster qui possède deux instances Amazon EC2 actives enregistrées, une instance `c4.4xlarge` et une instance `c4.large`. L'instance `c4.4xlarge` s'enregistre dans le cluster avec `16,384` unités d'UC et `30,158` Mio de mémoire. L'instance `c4.large` s'enregistre avec `2,048` unités d'UC et `3,768` Mio de mémoire. Les ressources globales de ce cluster sont de `18,432` unités d'UC et de `33,926` Mio de mémoire.

Si dix tâches sont en cours d'exécution sur ce cluster et que chaque tâche consomme `1,024` unités d'UC et `2,048` Mio de mémoire, un total de `10,240` unités d'UC et `20,480` Mio de mémoire est utilisé sur le cluster. Ceci est rapporté à CloudWatch soit 55 % d'utilisation du processeur et 60 % d'utilisation de la mémoire pour le cluster.

Utilisation du système de fichiers Amazon EBS au niveau du cluster Amazon ECS

La métrique d'utilisation du système de fichiers EBS au niveau du cluster est calculée en divisant la quantité totale du système de fichiers EBS utilisé par les tâches exécutées sur le cluster par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches du cluster.

```
                                    (Total GB of EBS filesystem used by tasks in
 cluster x 100)
Cluster EBS filesystem utilization =
 ------------------------------------------------------------------
                                    (Total GB of EBS filesystem allocated to tasks
 in cluster)
```

Métriques d’utilisation d’un service Amazon ECS

Les mesures d’utilisation du service sont disponibles pour l’UC, pour la mémoire et, lorsqu’un volume EBS est associé à vos tâches, pour l’utilisation du système de fichiers EBS. Les métriques de niveau de service sont prises en charge pour les services dont les tâches sont hébergées à la fois sur des instances Amazon EC2 et Fargate.

Niveau de service : utilisation de l’UC et de la mémoire

L’utilisation de l’UC et de la mémoire est mesurée en pourcentage d’UC et de mémoire utilisés par les tâches Amazon ECS appartenant à un service sur un cluster, par rapport à l’UC et à la mémoire spécifiés dans la définition de tâche du service.

Lorsque vous consultez ces statistiques dans CloudWatch, vous pouvez choisir différentes statistiques :

- Moyenne : taux d’utilisation moyen pour toutes les tâches du service. Ceci est calculé à l'aide de la formule suivante.
- Minimum : utilisation de la tâche dont l’utilisation des ressources est la plus faible dans le service. Cela représente le pourcentage d’UC ou de mémoire utilisé par la tâche la moins gourmande en ressources par rapport à ce qui a été spécifié dans la définition de la tâche.
- Minimum : utilisation de la tâche dont l’utilisation des ressources est la plus faible dans le service. Cela représente le pourcentage d’UC ou de mémoire utilisé par la tâche la plus gourmande en ressources par rapport à ce qui a été spécifié dans la définition de la tâche.

Les formules suivantes montrent comment la statistique moyenne est calculée :

```
                                (Total CPU units used by tasks in service) x 100
Service CPU utilization =
 ------------------------------------------------------------
                          (Total CPU units specified in the task definition) x (Number
 of tasks in the service)
```

```
                                   (Total MiB of memory used by tasks in service x
 100)
Service memory utilization =
 ------------------------------------------------------------------
                             (Total MiB of memory specified in the task definition)  x
 (Number of tasks in the service)
```

> **Note**
>
> Les formules précédentes s'appliquent uniquement à la statistique moyenne. Les statistiques minimale et maximale représentent la tâche individuelle dont l'utilisation des ressources est la plus faible et la plus élevée, respectivement, plutôt qu'un calcul agrégé pour toutes les tâches.

Amazon ECS collecte des métriques toutes les 20 secondes. Chaque minute, l'agent de conteneur Amazon ECS calcule le nombre d'unités d'UC et la quantité de mémoire (en Mio) actuellement utilisées pour chaque tâche en cours d'exécution appartenant au service. Ces informations sont communiquées à Amazon ECS. La quantité totale de ressources processeur et mémoire utilisées par toutes les tâches du service en cours d'exécution sur le cluster est calculée, et ces chiffres sont transmis à CloudWatch sous la forme d'un pourcentage du total des ressources spécifiées pour le service dans la définition de tâche du service. Les valeurs minimale et maximale sont les plus petites et les plus grandes des métriques de 20 secondes. La moyenne est l'agrégat des 3 valeurs.

Si vous spécifiez une limite flexible (`memoryReservation`), elle est utilisée pour calculer la quantité de mémoire réservée. Sinon, la limite stricte (`memory`) est utilisée. Pour plus d'informations sur les limites strictes et souples, consultez la section Taille de la tâche.

Prenons l'exemple d'un service pour lequel la définition de tâche spécifie un total de 512 unités d'UC et 1 024 Mio de mémoire (avec le paramètre de limite stricte `memory`) pour l'ensemble de ses conteneurs. Le service a un nombre souhaité de 1 tâche en cours d'exécution, il s'exécute sur un cluster avec 1 instance de conteneur `c4.large` (avec 2 048 unités d'UC et 3 768 Mio de mémoire totale) et il n'y a pas d'autre tâche en cours d'exécution sur le cluster. Bien que la tâche spécifie 512 unités d'UC, comme il s'agit de la seule tâche en cours d'exécution sur une instance de conteneur comportant 2 048 unités d'UC, elle peut utiliser jusqu'à quatre fois la quantité spécifiée (2 048/512). Toutefois, la mémoire spécifiée de 1 024 Mio est une limite stricte et elle ne peut pas être dépassée. Ainsi, dans ce cas, l'utilisation de la mémoire par le service ne peut pas dépasser 100 %.

Si l'exemple précédent utilise la limite flexible `memoryReservation` au lieu du paramètre de limite stricte `memory`, les tâches du service peuvent, au besoin, utiliser plus de mémoire que les 1 024 Mio spécifiés. Dans ce cas, l'utilisation de la mémoire du service peut dépasser 100 %.

Si votre application connaît un pic soudain d'utilisation de la mémoire pendant une courte période, vous ne verrez pas l'utilisation de la mémoire du service augmenter car Amazon ECS collecte

plusieurs points de données par minute, puis les regroupe en un seul point de données envoyé à CloudWatch.

Si cette tâche est exécutée CPU-intensive fonctionnent pendant une période donnée et en utilisant les 2 048 unités centrales disponibles et 512 Mo de mémoire, le service indique une utilisation du processeur de 400 % et une utilisation de la mémoire de 50 %. Si la tâche est inactive et utilise 128 unités d'UC et 128 Mio de mémoire, le service signale 25 % d'utilisation de l'UC et 12,5 % d'utilisation de la mémoire.

> **Note**
>
> Dans cet exemple, l'utilisation de l'UC ne dépasse 100 % que lorsque les unités du processeur sont définies au niveau du conteneur. Si vous définissez des unités d'UC au niveau de la tâche, l'utilisation ne dépasse pas la limite définie au niveau de la tâche.

Utilisation du système de fichiers EBS au niveau de service

Le niveau de service d'utilisation du système de fichiers EBS est mesuré comme la quantité totale du système de fichiers EBS utilisée par les tâches appartenant au service, divisée par la quantité totale de stockage du système de fichiers EBS allouée à toutes les tâches appartenant au service.

Nombre de tâches de Service LIVE

Avec le `LiveTaskCount` métrique, vous obtenez une vue complète des tâches de votre service qui s'exécutent activement et génèrent des métriques. Il compte les tâches du `ACTIVATING`, `RUNNING`, et `DEACTIVATING` états, y compris les tâches en cours d'exécution mais en transition vers un état arrêté, telles que les tâches qui épuisent les connexions lors d'un événement de mise à l'échelle.

Ce comportement rend la métrique bien adaptée aux alarmes de dimensionnement automatique et à la surveillance de la capacité, car une tâche qui continue de servir du trafic alors qu'elle draine est toujours comptabilisée.

`LiveTaskCount` est disponible gratuitement. Pour plus d'informations, consultez Amazon ECS CloudWatch indicateurs.

Service **`COURIR`** nombre de tâches

Vous pouvez utiliser CloudWatch des statistiques pour afficher le nombre de tâches de vos services qui figurent dans le `RUNNING` état. Par exemple, vous pouvez définir un CloudWatch alarme pour

cette métrique afin de vous avertir si le nombre de tâches en cours d'exécution dans votre service tombe en dessous d'une valeur spécifiée.

Le `RunningTaskCount` la métrique compte les tâches dont l'état actuel et l'état souhaité sont à la fois `RUNNING`. Cela en fait un signal de santé et de stabilité pour votre service. Il ne compte pas les tâches qui sont toujours en cours d'exécution alors qu'elles passent à l'état arrêté. Pour compter également ces tâches, utilisez `LiveTaskCount`. Pour plus d'informations, consultez Nombre de tâches de Service LIVE.

Service **`COURIR`** nombre de tâches dans Amazon ECS CloudWatch Informations sur les conteneurs

Un « certain nombre de tâches en cours » (`RunningTaskCount`) la métrique est disponible par cluster et par service lorsque vous utilisez Amazon ECS CloudWatch Informations sur les conteneurs. Vous pouvez utiliser Container Insights pour tous les nouveaux clusters créés en accédant au `containerInsights` réglage du compte, sur des clusters individuels en activant les paramètres du cluster lors de la création du cluster, ou sur des clusters existants à l'aide du UpdateClusterSettings API.

> **Note**
>
> À compter du 1er octobre 2026, vous ne serez pas facturé pour le `RunningTaskCount` métrique en CloudWatch Container Insights, quel que soit le nombre de services ou de clusters Amazon ECS sur votre compte.

Autres indicateurs collectés par CloudWatch Container Insights est facturé en tant que mesures personnalisées. Pour plus d'informations sur CloudWatch tarification, voir CloudWatch Tarification.

Pour consulter cette métrique, consultez Statistiques d'Amazon ECS Container Insights dans le Amazon CloudWatch Guide de l'utilisateur.

High-resolution indicateurs de service

Par défaut, Amazon ECS publie `CPUUtilization` et `MemoryUtilization` mesures de service pour CloudWatch à une résolution de 60 secondes. Vous pouvez configurer une résolution de 20 secondes en définissant une configuration de surveillance sur votre service. Cela permet de détecter plus rapidement les changements d'utilisation des ressources et peut être utilisé pour accélérer la mise à l'échelle automatique des services.

Pour activer les métriques haute résolution, spécifiez une configuration de surveillance avec `resolutionSeconds` réglé sur `20` lors de la création ou de la mise à jour de votre service. Pour obtenir les instructions de configuration complètes, y compris la configuration d'une politique de mise à l'échelle haute résolution, voir Mise à l'échelle automatique plus rapide grâce à des mesures haute résolution.

Cas d’utilisation des métriques d’utilisation d’un service Amazon ECS

La liste suivante fournit des informations sur les circonstances dans lesquelles utiliser les métriques Amazon ECS.

- Surveillance de l’utilisation des ressources : utilisez des statistiques de moyenne pour surveiller les modèles de consommation de l’UC et de la mémoire, établir des bases de performance et détecter les tendances de dégradation progressive des performances.
- Optimisation des coûts : utilisez des statistiques de moyenne pour surveiller l’utilisation des ressources, dimensionner correctement les conteneurs, ajuster les réserves en fonction des modèles d’utilisation et mettre en œuvre une mise à l’échelle planifiée.
- Analyse comparative des performances : utilisez des statistiques de moyenne pour comparer les métriques entre les révisions de service, établir des KPI de performance et valider les améliorations d’optimisation.
- Détection du plancher de ressources : utilisez des statistiques de moyenne pour identifier les besoins en ressources minimales pendant les périodes d’inactivité, définir les réserves appropriées et détecter les baisses anormales.
- Détection des anomalies : utilisez des statistiques de minimum pour détecter les baisses inhabituelles d’utilisation des ressources qui indiquent des problèmes potentiels, tels que des échecs d’initialisation ou des périodes d’inactivité inattendues.
- Affinement de la politique de mise à l’échelle : utilisez des statistiques de minimum pour établir des seuils de mise à l’échelle optimaux sur la base d’une utilisation viable minimale afin d’éviter une mise à l’échelle agressive.
- Planification des capacités : utilisez les statistiques de maximum pour définir des tailles de tâches appropriées et planifier une infrastructure avec une marge suffisante pour faire face aux pics de trafic.
- Identification des goulots d’étranglement en matière de performances : utilisez les statistiques de maximum pour détecter les points de saturation des ressources, identifier les goulots d’étranglement et déterminer quand augmenter la taille des tâches.

- Configuration de la politique de mise à l'échelle : utilisez les statistiques de maximum pour définir des seuils d'augmentation horizontale optimaux en fonction des modèles de pic et configurer la capacité de débordement.
- Surveillance de la conformité aux SLA : utilisez les statistiques de maximum pour suivre les temps de réponse de pointe et les taux d'erreur afin de garantir que les performances du service répondent aux SLA définis.

Informations sur les métriques connexes

Pour plus d'informations sur Container Insights, voir Exemples d'utilisation de Container Insights dans le CloudWatch Guide de l'utilisateur.

Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge

Utilisation d'Amazon EventBridge, vous pouvez automatiser votre AWS services et répondez automatiquement aux événements du système tels que les problèmes de disponibilité des applications ou les modifications de ressources. Événements de AWS les services sont fournis à EventBridge en temps quasi réel. Vous pouvez écrire des règles simples pour préciser les événements qui vous intéressent et les actions automatisées à effectuer quand un événement correspond à une règle. Les actions pouvant être configurées automatiquement sont les suivantes :

- Ajouter des événements dans lesquels enregistrer des groupes CloudWatch Journaux
- Invoquer un AWS Lambda fonction
- Appel de la fonctionnalité Exécuter la commande d'Amazon EC2
- Relais de l'événement à Amazon Kinesis Data Streams
- Activation d'un AWS Step Functions machine à états
- Notification d'une rubrique Amazon SNS ou d'une file d'attente Amazon Simple Queue Service (Amazon SQS)

Pour plus d'informations, voir Premiers pas avec Amazon EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

Vous pouvez utiliser les événements Amazon ECS pour EventBridge pour recevoir des notifications en temps quasi réel concernant l'état actuel de vos clusters Amazon ECS. Si elles utilisent EC2, vous

pouvez voir l'état des instances de conteneur et l'état actuel de toutes les tâches exécutées sur ces instances. Si vos tâches utilisent Fargate, vous pouvez voir l'état des instances de conteneur.

En utilisant EventBridge, vous pouvez créer des planificateurs personnalisés sur Amazon ECS qui sont chargés d'orchestrer les tâches entre les clusters et de surveiller l'état des clusters en temps quasi réel. Vous pouvez éliminer le code de planification et de surveillance qui interroge en permanence le service Amazon ECS pour détecter les changements d'état et gérer les changements d'état Amazon ECS de manière asynchrone à l'aide de n'importe quel EventBridge cible. Les cibles peuvent inclure AWS Lambda, Amazon Simple Queue Service, Amazon Simple Notification Service ou Amazon Kinesis Data Streams.

Un flux d'événements Amazon ECS garantit que chaque événement est diffusé au moins une fois. Si des événements dupliqués sont envoyés, l'événement fournit suffisamment d'informations pour identifier les doublons. Pour plus d'informations, consultez Gestion des événements Amazon ECS.

Les événements sont relativement classés, afin que vous puissiez facilement savoir lorsqu'un événement s'est produit en rapport avec d'autres événements.

Rubriques

- Événements Amazon ECS
- Gestion des événements Amazon ECS

Événements Amazon ECS

Amazon ECS suit l'état de chacune de vos tâches et de chacun de vos services. Si l'état d'une tâche ou d'un service change, un événement est généré et envoyé à Amazon EventBridge. Ces événements sont considérés comme des événements de changement d'état de tâche et des événements d'action de service. Ces événements et leurs causes possibles sont détaillés dans les sections suivantes.

Amazon ECS génère et envoie les types d'événements suivants à EventBridge:

- Changement d'état d'une instance de conteneur
- Changement d'état d'une tâche
- Deployment state change (Changement d'état du déploiement)
- Changement d'état du crochet
- Action de service

- Modification de l'état de l'instance de conteneur
- Modification de l'état de déploiement du démon
- Action de service du démon

> **Note**
>
> Amazon ECS est susceptible d'ajouter d'autres types , sources et détails d'événements ultérieurement. Si vous désérialisez les données JSON d'événement dans le code, veillez à ce que votre application soit prête à gérer des propriétés inconnues, afin d'éviter des problèmes si et lorsque ces propriétés supplémentaires sont ajoutées.

Dans certains cas, plusieurs événements sont générés pour la même activité. Par exemple, lorsqu'une tâche est démarrée sur une instance de conteneur, un événement de modification de l'état de tâche est généré pour la nouvelle tâche. Un événement de changement d'état d'instance de conteneur est généré pour prendre en compte le changement dans les ressources disponibles, comme le processeur, la mémoire et les ports disponibles, dans l'instance de conteneur. De même, si une instance de conteneur est mise hors service, des événements sont générés pour l'instance de conteneur, l'état de connexion de l'agent de conteneur et chaque tâche en cours d'exécution sur l'instance de conteneur.

Les événements de changement d'état de conteneur et de changement d'état de tâche contiennent deux champs `version` : un dans le corps principal de l'événement et l'autre dans l'objet `detail` de l'événement. Les différences entre ces deux champs sont décrites ci-dessous :

- Le champ `version` du corps de l'événement est défini sur `0` pour tous les événements. Pour plus d'informations sur EventBridge paramètres, voir AWS métadonnées des événements de service dans le Amazon EventBridge Guide de l'utilisateur.
- Le champ `version` dans l'objet `detail` de l'événement décrit la version de la ressource associée. Chaque fois qu'une ressource change d'état, cette version est incrémentée. Comme les événements peuvent être envoyés plusieurs fois, ce champ vous permet d'identifier les événements en double. Les événements en double présentent la même version dans l'objet `detail`. Si vous répliquez votre instance de conteneur Amazon ECS et l'état de vos tâches avec EventBridge, vous pouvez comparer la version d'une ressource signalée par les API Amazon ECS avec la version indiquée dans EventBridge pour la ressource (dans le `detail` object) pour vérifier que la version de votre flux d'événements est à jour.

Les événements d'action de service contiennent uniquement le champ `version` dans le corps principal.

Les événements d’action de service spécifient le service dans deux champs différents :

- Pour les événements générés par `create-service`, le service est dans le champ `serviceName`.
- Pour les événements générés par `update-service`, le service est dans le champ `service`.

Si vous utilisez des outils automatisés pour les événements de service, vous devez coder les deux champs.

Pour de plus amples informations sur la création d’une règle pour les événements d’action de service, consultez la section Événements d’action d’un service Amazon ECS.

Pour plus d'informations sur la manière d'intégrer Amazon ECS et EventBridge, voir Intégrer Amazon EventBridge et Amazon ECS.

Événements de changement d’état d’une instance de conteneur Amazon ECS

Les scénarios suivants provoquent des événements de changement d'état d'instance de conteneur :

Vous appelez les opérations d'API `StartTask`, `RunTask` ou `StopTask` directement ou avec la Console de gestion AWS ou les kits SDK.

> Le fait de placer ou d'arrêter des tâches sur une instance de conteneur a pour effet de modifier les ressources disponibles sur l'instance de conteneur, comme l'UC, la mémoire et les ports disponibles.

Le planificateur de service Amazon ECS service démarre ou arrête une tâche.

> Le fait de placer ou d'arrêter des tâches sur une instance de conteneur a pour effet de modifier les ressources disponibles sur l'instance de conteneur, comme l'UC, la mémoire et les ports disponibles.

L'agent de conteneur Amazon ECS appelle l'opération d'API `SubmitTaskStateChange` avec un état `STOPPED` pour une tâche ayant l'état souhaité `RUNNING`.

> L'agent de conteneur Amazon ECS supervise l'état des tâches sur vos instances de conteneur, et il signale tout changement d'état. Si une tâche supposée être à l'état `RUNNING` passe à l'état `STOPPED`, l'agent libère les ressources qui ont été allouées à la tâche arrêtée, comme l'UC, la mémoire et les ports disponibles.

Vous désenregistrez l'instance de conteneur à l'aide du `DeregisterContainerInstance` Fonctionnement de l'API, soit directement, soit avec Console de gestion AWS ou des kits de développement logiciel.

L'annulation de l'enregistrement d'une instance de conteneur modifie l'état de l'instance de conteneur et l'état de connexion de l'agent de conteneur Amazon ECS.

Une tâche s'est arrêtée en même temps qu'une instance EC2.

Lorsque vous arrêtez une instance de conteneur, les tâches qui sont en cours d'exécution sur elle passe à l'état `STOPPED`.

L'agent de conteneur Amazon ECS enregistre une instance de conteneur pour la première fois.

La première fois que l'agent de conteneur Amazon ECS enregistre une instance de conteneur (lors du lancement ou de la première exécution manuelle), cela crée un événement de changement d'état pour l'instance.

L'agent de conteneur Amazon ECS se connecte à Amazon ECS ou s'en déconnecte.

Lorsque l'agent de conteneur Amazon ECS se connecte ou se déconnecte du backend Amazon ECS, il modifie l'état `agentConnected` de l'instance de conteneur.

> **Note**
>
> L'agent de conteneur Amazon ECS se déconnecte et se reconnecte plusieurs fois par heure dans le cadre de son fonctionnement normal, si bien que des événements de connexion de l'agent sont prévisibles. Ces événements ne signifient pas qu'il existe un problème au niveau de l'agent de conteneur ou de votre instance de conteneur.

Vous mettez à niveau l'agent de conteneur Amazon ECS sur une instance.

Le détail de l'instance de conteneur contient un objet pour la version de l'agent de conteneur. Si vous mettez à niveau l'agent, les informations de cette version changent et génèrent un événement.

ExampleÉvénement de changement d'état d'instance de conteneur

Les événements de changement d'état d'instance de conteneur sont remis dans le format suivant. Le `detail` section ressemble à ContainerInstance objet renvoyé par un DescribeContainerInstances

Fonctionnement de l'API dans Référence de l'API Amazon Elastic Container Service. Pour plus d'informations sur EventBridge paramètres, voir AWS métadonnées des événements de service dans le Amazon EventBridge Guide de l'utilisateur.

```
{
  "version": "0",
  "id": "8952ba83-7be2-4ab5-9c32-6687532d15a2",
  "detail-type": "ECS Container Instance State Change",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2016-12-06T16:41:06Z",
  "region": "us-east-1",
  "resources": [
    "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315"
  ],
  "detail": {
    "agentConnected": true,
    "attributes": [
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role-network-host"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.awslogs"
      },
      {
        "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
      },
      {
        "name": "com.amazonaws.ecs.capability.privileged-container"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
      },
```

```
      {
        "name": "com.amazonaws.ecs.capability.ecr-auth"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.20"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.21"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.22"
      },
      {
        "name": "com.amazonaws.ecs.capability.docker-remote-api.1.23"
      },
      {
        "name": "com.amazonaws.ecs.capability.task-iam-role"
      }
    ],
    "clusterArn": "arn:aws:ecs:us-east-1:111122223333:cluster/default",
    "containerInstanceArn": "arn:aws:ecs:us-east-1:111122223333:container-instance/
b54a2a04-046f-4331-9d74-3f6d7f6ca315",
    "ec2InstanceId": "i-f3a8506b",
    "registeredResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 2048
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 3767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
```

```
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "remainingResources": [
      {
        "name": "CPU",
        "type": "INTEGER",
        "integerValue": 1988
      },
      {
        "name": "MEMORY",
        "type": "INTEGER",
        "integerValue": 767
      },
      {
        "name": "PORTS",
        "type": "STRINGSET",
        "stringSetValue": [
          "22",
          "2376",
          "2375",
          "51678",
          "51679"
        ]
      },
      {
        "name": "PORTS_UDP",
        "type": "STRINGSET",
        "stringSetValue": []
      }
    ],
    "status": "ACTIVE",
    "version": 14801,
    "versionInfo": {
      "agentHash": "aebcbca",
      "agentVersion": "1.13.0",
      "dockerVersion": "DockerVersion: 1.11.2"
    },
    "updatedAt": "2016-12-06T16:41:06.991Z"
  }
```

```
}
```

Événements de changement d’état d’une tâche Amazon ECS

Les scénarios suivants provoquent des événements de changement d'état de tâche :

Vous appelez les opérations d'API `StartTask`, `RunTask` ou `StopTask` directement ou avec Console de gestion AWS, la AWS CLI ou les kits SDK.

Le démarrage ou l'arrêt de tâches crée de nouvelles ressources de tâche ou modifie l'état des ressources de tâche existantes.

Le planificateur de service Amazon ECS service démarre ou arrête une tâche.

Le démarrage ou l'arrêt de tâches crée de nouvelles ressources de tâche ou modifie l'état des ressources de tâche existantes.

L'agent de conteneur Amazon ECS appelle l'opération d'API `SubmitTaskStateChange`.

Pour EC2, l’agent de conteneur Amazon ECS surveille l’état de vos tâches sur vos instances de conteneur. L’agent de conteneur Amazon ECS signale tout changement d’état. L'état peut notamment passer de `PENDING` à `RUNNING` ou de `RUNNING` à `STOPPED`.

Vous forcez le désenregistrement de l'instance de conteneur sous-jacente à l'aide du `DeregisterContainerInstance` Fonctionnement de l'API et `force` drapeau, soit directement, soit avec Console de gestion AWS ou des kits de développement logiciel.

L'annulation de l'enregistrement d'une instance de conteneur modifie l'état de l'instance de conteneur et l'état de connexion de l'agent de conteneur Amazon ECS. Si les tâches sont en cours d'exécution sur l'instance de conteneur, l'indicateur `force` doit être défini pour permettre l'annulation de l'enregistrement. Cette action arrête toutes les tâches sur l'instance.

L'instance de conteneur sous-jacente est arrêtée ou résiliée.

Lorsque vous arrêtez ou résiliez une instance de conteneur, les tâches qui sont en cours d'exécution sur cette dernière passe à l'état `STOPPED`.

L'état du conteneur de la tâche est modifié.

L'agent de conteneur Amazon ECS supervise l'état des conteneurs dans les tâches. Par exemple, si un conteneur en cours d'exécution dans une tâche s'arrête, le changement d'état de ce dernier génère un événement.

Une tâche utilisant le fournisseur de capacité Fargate Spot reçoit un avis de résiliation.

Lorsqu'une tâche utilise le fournisseur de capacité `FARGATE_SPOT` et qu'elle est arrêtée en raison d'une interruption Spot, un événement de changement d'état de tâche est généré.

ExampleÉvénement de modification de l'état de la tâche

Les événements de changement d'état de tâche sont remis dans le format suivant. Remarquez les points suivants concernant les champs :

- L'état de l'événement n'est pas disponible dans l'événement de changement d'état d'une tâche. Si vous avez besoin de l'état des tâches, vous pouvez exécuter describe-tasks.
- Si vos conteneurs utilisent une image hébergée avec Amazon ECR, le champ `imageDigest` est renvoyé.
- Les valeurs des champs `createdAt`, `connectivityAt`, `pullStartedAt`, `startedAt`, `pullStoppedAt` et `updatedAt` sont des horodatages de chaînes ISO.
- La valeur de `detail-type` est « Changement d'état de tâche ECS ».
- Lorsque l'événement est généré pour une tâche arrêtée, les champs `stoppedReason` et `stopCode` fournissent des informations supplémentaires sur la raison pour laquelle la tâche s'est arrêtée (par exemple, « Initié par l'utilisateur »).

Pour plus d'informations sur EventBridge paramètres, voir AWS métadonnées des événements de service dans le Amazon EventBridge Référence des événements.

Pour plus d'informations sur la configuration d'un Amazon EventBridge règle événementielle qui capture uniquement les événements de tâche lorsque la tâche a cessé de s'exécuter parce que l'un de ses conteneurs essentiels s'est arrêté, voir Envoi d'alertes Amazon Simple Notification Service pour les événements de tâche Amazon ECS arrêtés

```
{
    "version": "0",
    "id": "105f6bb1-4da6-c630-4965-35383018cbca",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-06T11:02:34Z",
    "region": "us-east-1",
    "resources": [
```

```
        "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/a1173316d40a45dea9"
    ],
    "detail": {
        "attachments": [
            {
                "id": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0103a2f01bad57d71"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:50:d1:c1:77:81"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-163.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.163"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "FARGATE",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example-cluster",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-06T11:02:17.19Z",
        "containers": [
```

```
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a0a99b87-baa8-4bf6-b9f1-a9a95917a635",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId": "a1173316d40a45dea9-0265927825",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/a2010e2d-ba7c-4135-8b79-e0290ff3cd8c",
                "lastStatus": "RUNNING",
                "name": "aws-guardduty-agent-nm40lC",
                "imageDigest":
 "sha256:bf9197abdf853607e5fa392b4f97ccdd6ca56dd179be3ce8849e552d96582ac8",
                "runtimeId": "a1173316d40a45dea9-2098416933",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "null"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example-
cluster/a1173316d40a45dea9/dccf0ca2-d929-471f-a5c3-98006fd4379e",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
```

```
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId": "a1173316d40a45dea9-4027662657",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
                "networkInterfaces": [
                    {
                        "attachmentId": "fe3a9a46-6a47-40ee-afd9-7952ae90a75a",
                        "privateIpv4Address": "10.0.1.163"
                    }
                ],
                "cpu": "0"
            }
        ],
        "cpu": "256",
        "createdAt": "2025-05-06T11:02:13.877Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "ephemeralStorage": {
            "sizeInGiB": 20
        },
        "group": "family:webserver",
        "launchType": "FARGATE",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "web"
                },
                {
                    "environment": [
                        {
                            "name": "CLUSTER_NAME",
                            "value": "example-cluster"
                        },
                        {
                            "name": "REGION",
                            "value": "us-east-1"
                        },
                        {
                            "name": "HOST_PROC",
                            "value": "/host_proc"
                        },
```

```
                        {
                            "name": "AGENT_RUNTIME_ENVIRONMENT",
                            "value": "ecsfargate"
                        },
                        {
                            "name": "STAGE",
                            "value": "prod"
                        }
                    ],
                    "memory": 128,
                    "name": "aws-guardduty-agent-nm40lC"
                },
                {
                    "name": "aws-otel-collector"
                }
            ]
        },
        "platformVersion": "1.4.0",
        "pullStartedAt": "2025-05-06T11:02:24.162Z",
        "pullStoppedAt": "2025-05-06T11:02:33.493Z",
        "startedAt": "2025-05-06T11:02:34.325Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example-cluster/
a1173316d40a45dea9",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver:5",
        "updatedAt": "2025-05-06T11:02:34.325Z",
        "version": 3
    }
}
```

Example

Voici un exemple d'événement de changement d'état d'une tâche pour EC2.

```
{
    "version": "0",
    "id": "a65cf262-f104-0dd5-ceda-4b09ba71a441",
    "detail-type": "ECS Task State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2025-05-12T13:12:06Z",
    "region": "us-east-1",
    "resources": [
        "arn:aws:ecs:us-east-1:123456789012:task/example/c1ffa94f19a540ed8d9f7e1d2a5d"
```

```
    ],
    "detail": {
        "attachments": [
            {
                "id": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                "type": "eni",
                "status": "ATTACHED",
                "details": [
                    {
                        "name": "subnetId",
                        "value": "subnet-0d0eab1bb38d5ca64"
                    },
                    {
                        "name": "networkInterfaceId",
                        "value": "eni-0ea90f746500773a4"
                    },
                    {
                        "name": "macAddress",
                        "value": "0e:d5:9b:ce:49:fb"
                    },
                    {
                        "name": "privateDnsName",
                        "value": "ip-10-0-1-37.ec2.internal"
                    },
                    {
                        "name": "privateIPv4Address",
                        "value": "10.0.1.37"
                    }
                ]
            }
        ],
        "attributes": [
            {
                "name": "ecs.cpu-architecture",
                "value": "x86_64"
            }
        ],
        "availabilityZone": "us-east-1b",
        "capacityProviderName": "Infra-ECS-Cluster-example-fa84e0cc-
AsgCapacityProvider-OseQJU9pizmp",
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/example",
        "connectivity": "CONNECTED",
        "connectivityAt": "2025-05-12T13:11:44.98Z",
```

```
        "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-instance/
example/d1d84798400f49f3b21cb61610c1e",
        "containers": [
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/197d0994-5367-4a6d-9f9a-f075e4a6",
                "lastStatus": "RUNNING",
                "name": "aws-otel-collector",
                "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.32.0",
                "imageDigest":
 "sha256:7a1b3560655071bcacd66902c20ebe9a69470d5691fe3bd36baace7c2f3c4640",
                "runtimeId":
 "8e926f0ccd8fe2b459926f49584ba6d33a3d9f61398dbabe944ee6a13a8ff3a1",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "0"
            },
            {
                "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/example/
c1ffa94f19a540ed8d9f7e1d2a5d3626/cab39ef0-9c50-459d-844b-b9d51d73d",
                "lastStatus": "RUNNING",
                "name": "web",
                "image": "nginx",
                "imageDigest":
 "sha256:c15da6c91de8d2f436196f3a768483ad32c258ed4e1beb3d367a27ed67253e66",
                "runtimeId":
 "9f1c73f0094f051541d9e5c2ab1e172d83c4eb5171bcc857c4504b02770ff3b8",
                "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
                "networkInterfaces": [
                    {
                        "attachmentId": "52333e3b-b812-41a8-b057-9ed184bbe5e1",
                        "privateIpv4Address": "10.0.1.37"
                    }
                ],
                "cpu": "99",
                "memory": "100"
            }
```

```
        ],
        "cpu": "256",
        "createdAt": "2025-05-12T13:11:44.98Z",
        "desiredStatus": "RUNNING",
        "enableExecuteCommand": false,
        "group": "family:webserver",
        "launchType": "EC2",
        "lastStatus": "RUNNING",
        "memory": "512",
        "overrides": {
            "containerOverrides": [
                {
                    "name": "aws-otel-collector"
                },
                {
                    "name": "web"
                }
            ]
        },
        "pullStartedAt": "2025-05-12T13:11:59.491Z",
        "pullStoppedAt": "2025-05-12T13:12:05.896Z",
        "startedAt": "2025-05-12T13:12:06.053Z",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/example/
c1ffa94f19a540ed8d9f7e1d2a5d",
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/
webserver",
        "updatedAt": "2025-05-12T13:12:06.053Z",
        "version": 4
    }
}
```


Événements d’action d’un service Amazon ECS

Amazon ECS envoie des événements d'action de service avec le type de détail Action de service ECS. Contrairement aux événements de changement d'état d'instance de conteneur et de tâche, les événements d'action de service n'incluent pas de numéro de version dans le champ de réponse `details`. Le modèle d'événement suivant est utilisé pour créer un EventBridge règle pour les événements d'action du service Amazon ECS. Pour plus d'informations, voir Commencer avec EventBridge dans le Amazon EventBridge Guide de l'utilisateur.


```
{
```

```
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Service Action"
    ]
}
```

Amazon ECS envoie des événements avec les types d'événements `INFO`, `WARN` et `ERROR`. Les événements d'action de service sont les suivants.

Événements d'action de service avec **`INFO`** type d'événement

`SERVICE_STEADY_STATE`

Le service est sain et avec le nombre de tâches souhaité, atteignant ainsi un état stable. Le planificateur de service rapporte l'état de façon régulière, vous pouvez donc recevoir ce message plusieurs fois.

`TASKSET_STEADY_STATE`

L'ensemble de tâches est sain et avec le nombre de tâches souhaité, atteignant ainsi un état stable.

`CAPACITY_PROVIDER_STEADY_STATE`

Un fournisseur de capacité associé à un service atteint un état stable.

`SERVICE_DESIRED_COUNT_UPDATED`

Lorsque le planificateur de service met à jour le nombre souhaité calculé pour un service ou un ensemble de tâches. Cet événement n'est pas envoyé lorsque le nombre souhaité est mis à jour manuellement par un utilisateur.

`TASKS_STOPPED`

Le service a arrêté la tâche en cours d’exécution.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Le déploiement de service est en cours. Le déploiement de service peut être une restauration ou une nouvelle révision de service.

`SERVICE_DEPLOYMENT_COMPLETED`

Le déploiement de service est en état stable et est terminé. Le déploiement de service peut être une restauration ou une mise à jour du service.

Événements d'action de service avec **`AVERTIR`** type d'événement

`SERVICE_TASK_START_IMPAIRED`

Le service n'est pas en mesure de démarrer invariablement les tâches avec succès.

`SERVICE_DISCOVERY_INSTANCE_UNHEALTHY`

Un service utilisant la découverte de service contient une tâche non saine. Le planificateur de service détecte qu'une tâche au sein d'un registre de service n'est pas saine.

`VPC_LATTICE_TARGET_UNHEALTHY`

Le service utilisant VPC Lattice a détecté que l’une des cibles du VPC Lattice est défectueuse.

Événements d'action de service avec **`ERREUR`** type d'événement

`SERVICE_DAEMON_PLACEMENT_CONSTRAINT_VIOLATED`

Une tâche dans un service utilisant la stratégie du planificateur de service `DAEMON` n'est plus conforme à la stratégie de contrainte de placement du service.

`ECS_OPERATION_THROTTLED`

Le planificateur de service a été limité en raison des restrictions de l'API Amazon ECS.

`SERVICE_DISCOVERY_OPERATION_THROTTLED`

Le planificateur de services a été limité en raison de AWS Cloud Map Limites d'accélération de l'API. Cela peut se produire au niveau des services configurés pour utiliser la découverte de service.

`SERVICE_TASK_PLACEMENT_FAILURE`

Le planificateur de service ne parvient pas à placer une tâche. La cause est décrite dans le champ `reason`.

Une cause courante de la génération de cet événement de service est une quantité insuffisante de ressources dans le cluster pour pouvoir placer la tâche. Par exemple, la capacité de l'UC ou de la mémoire peut être insuffisante dans les instances de conteneur disponibles, ou aucune

instance de conteneur n'est disponible. Une autre cause fréquente est la déconnexion de l'agent de conteneur Amazon ECS dans l'instance de conteneur, ce qui empêche le planificateur de placer la tâche.

`SERVICE_TASK_CONFIGURATION_FAILURE`

Le planificateur de service ne parvient pas à placer une tâche en raison d'une erreur de configuration. La cause est décrite dans le champ `reason`.

L'une des causes courantes de la génération de cet événement de service est l'application de balises au service alors que l'utilisateur ou le rôle n'a pas activé le nouveau format Amazon Resource Name (ARN) dans la région. Pour plus d'informations, consultez Amazon Resource Names (ARN) et ID. Une autre cause fréquente est l'incapacité d'Amazon ECS à assumer le rôle IAM fourni pour la tâche.

`SERVICE_HEALTH_UNKNOWN`

Le service n'a pas été en mesure de décrire les données d'état relatives aux tâches.

`SERVICE_DEPLOYMENT_FAILED`

Le déploiement de service n'a pas atteint la stabilité. Cela se produit lorsqu'un CloudWatch est déclenché ou le disjoncteur détecte une panne de déploiement du service.

ExampleÉvénement d'état stable de service

Les événements d'état stable de service sont remis dans le format suivant. Pour plus d'informations sur EventBridge paramètres, voir Événements en EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

```
{
    "version": "0",
    "id": "af3c496d-f4a8-65d1-70f4-a69d52e9b584",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:27:22Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
```

```
        "eventName": "SERVICE_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:27:22.695Z"
    }
}
```

ExampleÉvénement d'état stable de fournisseur de capacité

Les événements d'état stable de fournisseur de capacité sont remis dans le format suivant.

```
{
    "version": "0",
    "id": "b9baa007-2f33-0eb1-5760-0d02a572d81f",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:37:00Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "INFO",
        "eventName": "CAPACITY_PROVIDER_STEADY_STATE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/ASG-tutorial-
capacity-provider"
        ],
        "createdAt": "2019-11-19T19:37:00.807Z"
    }
}
```

ExampleÉvénement de lancement de tâche de service défaillant

Les événements de lancement de tâche de service défaillant sont remis dans le format suivant.

```
{
    "version": "0",
    "id": "57c9506e-9d21-294c-d2fe-e8738da7e67d",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
```

```
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "WARN",
        "eventName": "SERVICE_TASK_START_IMPAIRED",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "createdAt": "2019-11-19T19:55:38.725Z"
    }
}
```

ExampleÉvénement d'échec de placement de tâche de service

Les événements d'échec de placement de tâche de service sont remis dans le format suivant. Pour plus d'informations, voir Événements en EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

Dans l'exemple suivant, la tâche tentait d'utiliser le fournisseur de capacité `FARGATE_SPOT`, mais le planificateur de service n'a pas pu acquérir de capacité Fargate Spot.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
    "detail-type": "ECS Service Action",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2019-11-19T19:55:38Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
        "eventType": "ERROR",
        "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
        "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
        "capacityProviderArns": [
            "arn:aws:ecs:us-west-2:111122223333:capacity-provider/FARGATE_SPOT"
        ],
        "reason": "RESOURCE:FARGATE",
        "createdAt": "2019-11-06T19:09:33.087Z"
    }
```

```
}
```

Dans l'exemple suivant pour EC2, la tâche a été lancée sur l'instance de conteneur `2dd1b186f39845a584488d2ef155c131`, mais le planificateur de service n'a pas pu placer la tâche en raison du manque de ressources d'UC.

```
{
  "version": "0",
  "id": "ddca6449-b258-46c0-8653-e0e3a6d0468b",
  "detail-type": "ECS Service Action",
  "source": "aws.ecs",
  "account": "111122223333",
  "time": "2019-11-19T19:55:38Z",
  "region": "us-west-2",
  "resources": [
    "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
  ],
  "detail": {
    "eventType": "ERROR",
    "eventName": "SERVICE_TASK_PLACEMENT_FAILURE",
    "clusterArn": "arn:aws:ecs:us-west-2:111122223333:cluster/default",
    "containerInstanceArns": [
    "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
    ],
    "reason": "RESOURCE:CPU",
    "createdAt": "2019-11-06T19:09:33.087Z"
  }
}
```

Événements de modification de l'état de déploiement de service Amazon ECS

Amazon ECS envoie des événements de modification d'état de déploiement de service avec le type de détail ECS Deployment State Change (Modification d'état de déploiement ECS). Le modèle d'événement suivant est utilisé pour créer un EventBridge règle pour les événements de changement d'état de déploiement du service Amazon ECS. Pour plus d'informations sur la création d'un EventBridge règle, voir Premiers pas avec Amazon EventBridge dans le Amazon EventBridge Guide de l'utilisateur.

```
{
    "source": [
```

```
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Deployment State Change"
    ]
}
```

Amazon ECS envoie des événements avec les types d'événements `INFO` et `ERROR`. Pour plus d'informations, consultez Événements d'action d'un service Amazon ECS

Vous trouverez ci-dessous des événements de modification de l'état de déploiement de service.

`SERVICE_DEPLOYMENT_IN_PROGRESS`

Le déploiement du service est en cours. Cet événement est envoyé pour les déploiements initiaux et les déploiements de restauration.

`SERVICE_DEPLOYMENT_COMPLETED`

Le déploiement du service est terminé. Cet événement est envoyé lorsqu'un service atteint un état stable après un déploiement.

`SERVICE_DEPLOYMENT_FAILED`

Le déploiement du service a échoué. Cet événement est envoyé pour les services avec la logique de disjoncteur de circuit de déploiement activée.

Example déploiement de service dans l'événement de progression

Le déploiement de service dans les événements de progression est fourni lors du démarrage d'un déploiement initial et de restauration. La différence entre les deux est dans le champ `reason`. Pour plus d'informations sur EventBridge paramètres, voir AWS métadonnées relatives aux événements de service ; dans le Amazon EventBridge Guide de l'utilisateur.

Vous trouverez ci-dessous un exemple de sortie pour le démarrage d'un déploiement initial.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3a6EXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
```

```
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment deploymentId in progress."
   }
}
```

Vous trouverez ci-dessous un exemple de sortie pour le démarrage d'un déploiement de restauration. Le champ `reason` fournit l'ID du déploiement vers lequel le service est en cours de restauration.

```
{
   "version": "0",
   "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
   "detail-type": "ECS Deployment State Change",
   "source": "aws.ecs",
   "account": "111122223333",
   "time": "2020-05-23T12:31:14Z",
   "region": "us-west-2",
   "resources": [
        "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
   ],
   "detail": {
        "eventType": "INFO",
        "eventName": "SERVICE_DEPLOYMENT_IN_PROGRESS",
        "deploymentId": "ecs-svc/123",
        "updatedAt": "2020-05-23T11:11:11Z",
        "reason": "ECS deployment circuit breaker: rolling back to
 deploymentId deploymentID."
   }
}
```

Exampleévénement de déploiement de service terminé

Les événements à l'état terminé de déploiement de service sont remis dans le format suivant. Pour plus d'informations, consultez Déploiement de services Amazon ECS en remplaçant des tâches.

```
{
```

```
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "INFO",
         "eventName": "SERVICE_DEPLOYMENT_COMPLETED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment deploymentID completed."
    }
}
```

Exampleévénement échoué de déploiement de service

Les événements à l'état échoué de déploiement de service sont remis dans le format suivant. Un événement à l'état échoué de déploiement de service n'est envoyé que pour les services dont la logique de disjoncteur de circuit de déploiement est activée. Pour plus d'informations, consultez Déploiement de services Amazon ECS en remplaçant des tâches.

```
{
    "version": "0",
    "id": "ddca6449-b258-46c0-8653-e0e3aEXAMPLE",
    "detail-type": "ECS Deployment State Change",
    "source": "aws.ecs",
    "account": "111122223333",
    "time": "2020-05-23T12:31:14Z",
    "region": "us-west-2",
    "resources": [
         "arn:aws:ecs:us-west-2:111122223333:service/default/servicetest"
    ],
    "detail": {
         "eventType": "ERROR",
         "eventName": "SERVICE_DEPLOYMENT_FAILED",
         "deploymentId": "ecs-svc/123",
         "updatedAt": "2020-05-23T11:11:11Z",
         "reason": "ECS deployment circuit breaker: task failed to start."
```

```
    }
}
```

Événements de changement d'état du hook de déploiement du service Amazon ECS

Amazon ECS envoie des événements de changement d'état du hook avec le type de détail Changement d'état du crochet ECS. Ces événements sont émis lorsqu'un hook du cycle de vie du déploiement change de statut. Le modèle d'événement suivant est utilisé pour créer un EventBridge règle pour les événements de changement d'état du hook Amazon ECS.

```
{
    "source": [
        "aws.ecs"
    ],
    "detail-type": [
        "ECS Hook State Change"
    ]
}
```

Les événements de changement d'état du hook sont les suivants.

`HOOK_IN_PROGRESS`

Un crochet est en cours.

`HOOK_AWAITING_ACTION`

Un crochet de pause a démarré et nécessite une action pour se terminer. Appel `ContinueServiceDeployment` pour poursuivre ou annuler le déploiement.

`HOOK_SUCCEEDED`

Le crochet s'est terminé avec succès. Pour les crochets de pause, cela signifie `ContinueServiceDeployment` a été appelé avec le `CONTINUE` action.

`HOOK_FAILED`

Le crochet a échoué. Pour les crochets de pause, cela signifie `ContinueServiceDeployment` a été appelé avec le `ROLLBACK` action.

`HOOK_TIMED_OUT`

Un hook a expiré sans être terminé.

Example crochet en attente d'un événement d'action

Ce qui suit montre un exemple d'événement lorsqu'un hook de pause démarre et attend une action.


```
{
    "version": "0",
    "id": "3329f79b-3dca-07f8-b1c2-5fe99f0b5e87",
    "detail-type": "ECS Hook State Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-03-05T15:54:41Z",
    "region": "us-west-2",
    "resources": ["arn:aws:ecs:us-west-2:123456789012:service-deployment/my-cluster/my-
service/0EYSiB0qap8xf0N76FsbE"],
    "detail": {
        "eventType": "INFO",
        "eventName": "HOOK_AWAITING_ACTION",
        "hookId": "ecs-pause-e7tK9G_WRJqNF_EOMjztDXfKenlJuEUVjsNStf4WLKw",
        "hookType": "PAUSE",
        "expiresAt": "2026-03-05T16:04:41Z",
        "lifecycleStage": "POST_TEST_TRAFFIC_SHIFT",
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "serviceArn": "arn:aws:ecs:us-west-2:123456789012:service/my-cluster/my-
service",
        "serviceDeploymentArn": "arn:aws:ecs:us-west-2:123456789012:service-deployment/
my-cluster/my-service/0EYSiB0qap8xf0N76FsbE",
        "updatedAt": "2026-03-05T15:54:41.618059641Z"
    }
}
```


Événements de modification de l'état de santé des instances de conteneurs Amazon ECS

Les scénarios suivants entraînent des événements de modification de l'état de l'instance de conteneur.

L'agent de conteneur Amazon ECS détecte une modification de l'état d'exécution du conteneur.

L'agent de conteneur évalue périodiquement l'état de santé du moteur d'exécution du conteneur (par exemple, le démon Docker) sur l'instance de conteneur. Si le moteur d'exécution ne répond plus ou se rétablit, l'agent signale la modification et un événement est généré.

Disponibilité : type de lancement EC2.

L'agent de conteneur Amazon ECS détecte une modification de l'état de santé accéléré des appareils de calcul.

Sur les instances dotées de périphériques de calcul accéléré tels que des GPU, l'agent de conteneur surveille l'état de santé des appareils. Si un appareil est endommagé ou se rétablit, un événement est généré.

Disponibilité : type de lancement des instances gérées Amazon ECS.

L'agent de conteneur Amazon ECS détecte une modification de l'état du démon.

L'agent de conteneur surveille les tâches de démon requises sur l'instance de conteneur. Si une tâche de démon échoue, un événement est généré.

Disponibilité : type de lancement des instances gérées Amazon ECS.

L'agent de conteneur Amazon ECS perd sa connectivité au plan de contrôle Amazon ECS.

Amazon ECS surveille en permanence la connectivité entre l'agent de conteneur et le plan de contrôle Amazon ECS. Lorsqu'une instance de conteneur reste déconnectée au-delà d'un certain seuil, Amazon ECS marque l'instance comme étant altérée.

Disponibilité : instances gérées Amazon ECS, AWS Fargate, et les types de lancement d'Amazon EC2.

L'état de santé général de l'instance de conteneur change.

Amazon ECS calcule un état de santé global à partir des résultats des bilans de santé individuels. Lorsque l'état général change, un événement est généré.

Disponibilité : instances gérées Amazon ECS et type de lancement EC2.

Exemple d'événement de modification de l'état d'une instance de conteneur

Les événements de modification de l'état de santé des instances de conteneurs sont fournis dans le format suivant. Pour plus d'informations sur les paramètres, voir AWS métadonnées relatives aux événements de service dans le Guide de l'utilisateur.

Exemple d'événement d'intégrité lors de l'exécution du conteneur

```
{
    "version": "0",
```

```
    "id": "EXAMPLE-48f8-c0ff-0dab-EXAMPLE2ac34",
    "detail-type": "ECS Container Instance Health Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-04-11T00:49:09Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:123456789012:container-instance/my-cluster/
a1b2c3d4EXAMPLE"
    ],
    "detail": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-instance/
my-cluster/a1b2c3d4EXAMPLE",
        "capacityProviderName": null,
        "ec2InstanceId": "i-0abcdef1234567890",
        "overallStatus": "OK",
        "healthChecks": [
            {
                "type": "CONTAINER_RUNTIME",
                "status": "OK",
                "lastStatusChange": "2026-04-11T00:48:53Z",
                "lastUpdated": "2026-04-11T00:48:09Z",
                "statusReason": null
            }
        ]
    }
}
```

Exemple d'événement de santé informatique accéléré

```
{
    "version": "0",
    "id": "EXAMPLE-46c3-a48a-1bef-EXAMPLE72f3c",
    "detail-type": "ECS Container Instance Health Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-04-11T00:49:09Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:123456789012:container-instance/my-cluster/
a1b2c3d4EXAMPLE"
    ],
```

```
    "detail": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-instance/
my-cluster/a1b2c3d4EXAMPLE",
        "capacityProviderName": "my-capacity-provider",
        "ec2InstanceId": "i-0abcdef1234567890",
        "overallStatus": "IMPAIRED",
        "healthChecks": [
            {
                "type": "ACCELERATED_COMPUTE",
                "status": "IMPAIRED",
                "lastStatusChange": "2026-04-11T00:48:53Z",
                "lastUpdated": "2026-04-11T00:48:09Z",
                "statusReason": "XID_46"
            }
        ]
    }
}
```

Exemple d'événement relatif à la santé d'un démon

```
{
    "version": "0",
    "id": "EXAMPLE-febc-b5e3-054b-EXAMPLE1a1f4f",
    "detail-type": "ECS Container Instance Health Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-04-11T00:49:09Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:123456789012:container-instance/my-cluster/
a1b2c3d4EXAMPLE"
    ],
    "detail": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-instance/
my-cluster/a1b2c3d4EXAMPLE",
        "capacityProviderName": "my-capacity-provider",
        "ec2InstanceId": "i-0abcdef1234567890",
        "overallStatus": "IMPAIRED",
        "healthChecks": [
            {
                "type": "DAEMON",
```

```
                "status": "IMPAIRED",
                "lastStatusChange": "2026-04-11T00:48:53Z",
                "lastUpdated": "2026-04-11T00:48:09Z",
                "statusReason": "Daemon task: EXAMPLE-4af2-4b3c-b9c1-EXAMPLEb2bbf is
 unhealthy"
            }
        ]
    }
}
```

Exemple d'événement de santé lié à la connectivité des agents

```
{
    "version": "0",
    "id": "EXAMPLE-7d2a-9f1c-4b6e-EXAMPLE9c8d1",
    "detail-type": "ECS Container Instance Health Change",
    "source": "aws.ecs",
    "account": "123456789012",
    "time": "2026-07-29T22:34:13Z",
    "region": "us-west-2",
    "resources": [
        "arn:aws:ecs:us-west-2:123456789012:container-instance/my-cluster/
a1b2c3d4EXAMPLE"
    ],
    "detail": {
        "clusterArn": "arn:aws:ecs:us-west-2:123456789012:cluster/my-cluster",
        "containerInstanceArn": "arn:aws:ecs:us-west-2:123456789012:container-instance/
my-cluster/a1b2c3d4EXAMPLE",
        "capacityProviderName": "my-capacity-provider",
        "ec2InstanceId": "i-0abcdef1234567890",
        "overallStatus": "IMPAIRED",
        "healthChecks": [
            {
                "type": "AGENT_CONNECTIVITY",
                "status": "IMPAIRED",
                "lastStatusChange": "2026-07-29T22:34:13Z",
                "lastUpdated": "2026-07-29T22:34:09Z",
                "statusReason": "Agent disconnected since 2026-07-29T22:34:13Z"
            }
        ]
    }
}
```

Gestion des événements Amazon ECS

Amazon ECS envoie des événements au moins une fois. Cela signifie que vous pouvez recevoir plusieurs copies d'un événement donné. En outre, des événements peuvent ne pas être transmis aux écouteurs d'événements dans l'ordre dans lequel ils se produisent.

Pour bien ordonner les événements, la section `detail` de chaque événement contient une propriété `version`. Chaque fois qu'une ressource change d'état, cette `version` est incrémentée. Les événements en double présentent la même `version` dans l'objet `detail`. Si vous répliquez votre instance de conteneur Amazon ECS et l'état de vos tâches avec EventBridge, vous pouvez comparer la version d'une ressource signalée par les API Amazon ECS avec `version` rapporté dans EventBridge pour que la ressource vérifie que la version de votre flux d'événements est à jour. Les événements avec un numéro de propriétés de version plus élevé doivent être traités comme postérieurs aux événements avec des numéros de version inférieurs.

Exemple : gestion des événements dans un AWS Lambda fonction

L'exemple suivant montre une fonction Lambda écrite en Python 3.9 qui capture les événements de changement d'état de tâche et d'instance de conteneur et les enregistre dans l'une des deux tables Amazon DynamoDB :

- ECSCtrInstanceState — Stocke le dernier état d'une instance de conteneur. L'ID de table correspond à la valeur `containerInstanceArn` de l'instance de conteneur.
- ECSTaskState — Stocke le dernier état d'une tâche. L'ID de table correspond à la valeur `taskArn` de la tâche.

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
```

```
      raise ValueError("Function only supports input from events with a source type
of: aws.ecs")

   # Switch on task/container events.
   table_name = ""
   if event["detail-type"] == "ECS Task State Change":
       table_name = "ECSTaskState"
       id_name = "taskArn"
       event_id = event["detail"]["taskArn"]
   elif event["detail-type"] == "ECS Container Instance State Change":
       table_name = "ECSCtrInstanceState"
       id_name =  "containerInstanceArn"
       event_id = event["detail"]["containerInstanceArn"]
   else:
       raise ValueError("detail-type for event is not a supported type. Exiting
without saving event.")

   new_record["cw_version"] = event["version"]
   new_record.update(event["detail"])

   # "status" is a reserved word in DDB, but it appears in containerPort
   # state change messages.
   if "status" in event:
       new_record["current_status"] = event["status"]
       new_record.pop("status")


   # Look first to see if you have received a newer version of an event ID.
   # If the version is OLDER than what you have on file, do not process it.
   # Otherwise, update the associated record with this latest information.
   print("Looking for recent event with same ID...")
   dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
   table = dynamodb.Table(table_name)
   saved_event = table.get_item(
       Key={
           id_name : event_id
       }
   )
   if "Item" in saved_event:
       # Compare events and reconcile.
       print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
       if saved_event["Item"]["version"] < event["detail"]["version"]:
           print("Received event is a more recent version than the stored event -
updating")
```

```
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
 ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

L'exemple Fargate suivant montre une fonction Lambda écrite en Python 3.9 qui capture les événements de changement d'état des tâches et les enregistre dans la table Amazon DynamoDB suivante :

```
import json
import boto3

def lambda_handler(event, context):
    id_name = ""
    new_record = {}

    # For debugging so you can see raw event format.
    print('Here is the event:')
    print((json.dumps(event)))

    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source type
 of: aws.ecs")

    # Switch on task/container events.
    table_name = ""
    if event["detail-type"] == "ECS Task State Change":
        table_name = "ECSTaskState"
        id_name = "taskArn"
        event_id = event["detail"]["taskArn"]
    else:
        raise ValueError("detail-type for event is not a supported type. Exiting
 without saving event.")

    new_record["cw_version"] = event["version"]
```

```
    new_record.update(event["detail"])

    # "status" is a reserved word in DDB, but it appears in containerPort
    # state change messages.
    if "status" in event:
        new_record["current_status"] = event["status"]
        new_record.pop("status")


    # Look first to see if you have received a newer version of an event ID.
    # If the version is OLDER than what you have on file, do not process it.
    # Otherwise, update the associated record with this latest information.
    print("Looking for recent event with same ID...")
    dynamodb = boto3.resource("dynamodb", region_name="us-east-1")
    table = dynamodb.Table(table_name)
    saved_event = table.get_item(
        Key={
            id_name : event_id
        }
    )
    if "Item" in saved_event:
        # Compare events and reconcile.
        print(("EXISTING EVENT DETECTED: Id " + event_id + " - reconciling"))
        if saved_event["Item"]["version"] < event["detail"]["version"]:
            print("Received event is a more recent version than the stored event -
updating")
            table.put_item(
                Item=new_record
            )
        else:
            print("Received event is an older version than the stored event -
ignoring")
    else:
        print(("Saving new event - ID " + event_id))

        table.put_item(
            Item=new_record
        )
```

Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée

Container Insights collecte, agrège et résume les mesures et les journaux de vos applications et microservices conteneurisés. Container Insights avec observabilité améliorée fournit toutes les métriques Container Insights, ainsi que des métriques supplémentaires sur les tâches et les conteneurs.

Container Insights détecte tous les conteneurs en cours d'exécution dans un cluster et collecte des données de performances à chaque couche de la pile de performances. Les données d'exploitation sont collectées en tant qu'événements de journaux de performances. Ces entrées utilisent un schéma JSON structuré qui permet aux données à haute cardinalité d'être intégrées et stockées à grande échelle. À partir de ces données, CloudWatch crée des mesures agrégées de niveau supérieur au niveau du cluster, du service et de la tâche comme CloudWatch métriques. Les métriques incluent l'utilisation des ressources telles que l'UC, la mémoire, le disque et le réseau. Les métriques sont disponibles dans les tableaux de bord CloudWatch automatiques.

Les métriques ne reflètent que les ressources avec des tâches en cours d'exécution pendant la période spécifiée. Par exemple, si vous avez un cluster avec un service mais que ce service n'a aucune tâche avec l'état `RUNNING`, aucune métrique n'est envoyée à CloudWatch. Si vous disposez de deux services et que l'un d'eux seulement a des tâches en cours d'exécution, seules les métriques du service avec des tâches en cours d'exécution seront envoyées.

Le 2 décembre 2024, AWS a publié Container Insights avec une observabilité améliorée pour Amazon ECS. Cette version prend en charge l'observabilité améliorée des clusters Amazon ECS utilisant Amazon EC2 et Fargate. Une fois que vous avez configuré Container Insights avec une observabilité améliorée sur Amazon ECS, Container Insights collecte automatiquement des données télémétriques détaillées sur l'infrastructure, du niveau du cluster jusqu'au niveau des conteneurs dans votre environnement, et affiche ces données de performances critiques dans des tableaux de bord organisés, ce qui vous évite d'avoir à effectuer des tâches fastidieuses pour configurer l'observabilité. Pour de plus amples informations sur la configuration de Container Insights avec observabilité améliorée, consultez la section Container Insights avec observabilité améliorée.

Nous vous recommandons d'utiliser Container Insights avec observabilité améliorée à la place de Container Insights, car il fournit une visibilité détaillée de votre environnement de conteneurs, réduisant ainsi le délai moyen de résolution. Pour plus d'informations, voir Amazon ECS Container Insights avec des mesures d'observabilité améliorées dans le Amazon CloudWatch Guide de l'utilisateur.

Les événements que vous pouvez consulter sont ceux qu'Amazon ECS envoie à Amazon EventBridge. Pour plus d'informations, consultez Événements Amazon ECS.

> **Important**
>
> À compter du 1er octobre 2026, aucune facture ne vous sera facturée `RunningTaskCount` métrique en CloudWatch Container Insights, quel que soit le nombre de services ou de clusters Amazon ECS sur votre compte.
> Autres indicateurs collectés par CloudWatch Container Insights est facturé en tant que mesures personnalisées. Pour plus d'informations sur CloudWatch tarification, voir CloudWatch Tarification.

Métriques supplémentaires pour les instances gérées Amazon ECS

Le tableau suivant répertorie les métriques supplémentaires disponibles pour les instances gérées Amazon ECS lors de l'utilisation de Container Insights.

Métrique	Description	Dimensions	Unit
`InstanceOSFilesystemUtilization`	Pourcentage de l'espace disque total utilisé (volume du système d'exploitation).	`ClusterName` - quand ContainerInsights est activé `ClusterName` , `CapacityProviderName` - quand ContainerInsights est activé `ClusterName` , `CapacityProviderName` , `ContainerInstanceId` , `EC2InstanceId` - quand EnhancedContainerInsights est activé	Pourcentage
`InstanceDataFilesystemUtilization`	Pourcentage de l'espace disque total utilisé (volume de données).	`ClusterName` - quand ContainerInsights est activé	Pourcentage

Métrique	Description	Dimensions	Unit
		`ClusterName` , `CapacityProviderName` - quand ContainerInsights est activé `ClusterName` , `CapacityProviderName` , `ContainerInstanceId` , `EC2InstanceId` - quand EnhancedContainerInsights est activé	
`InstanceGPULimit`	Nombre total de GPU disponibles sur l’instance. Disponible uniquement pour les instances gérées Amazon ECS exécutant NVIDIA GPU-enabled Types d'instances Amazon EC2.	`ClusterName` - quand EnhancedContainerInsights est activé `ClusterName` , `CapacityProviderName` - quand EnhancedContainerInsights est activé `ClusterName` , `CapacityProviderName` , `ContainerInstanceId` , `EC2InstanceId` - quand EnhancedContainerInsights est activé	Nombre

Métrique	Description	Dimensions	Unit
`InstanceGPUUsageTotal`	Nombre de GPU actuellement alloués aux tâches en cours d'exécution sur l'instance. Disponible uniquement pour les instances gérées Amazon ECS exécutant NVIDIA GPU-enabled Types d'instances Amazon EC2.	`ClusterName` - quand EnhancedContainerInsights est activé `ClusterName` , `CapacityProviderName` - quand EnhancedContainerInsights est activé `ClusterName` , `CapacityProviderName` , `ContainerInstanceId` , `EC2InstanceId` - quand EnhancedContainerInsights est activé	Nombre

Statistiques de redémarrage des applications GPU

Pour les instances gérées Amazon ECS exécutant NVIDIA GPU-enabled Types d'instances Amazon EC2, les `TaskGPURestartAppXidCount` et `ContainerGPURestartAppXidCount` les métriques comptent les erreurs NVIDIA Xid dont le compartiment de résolution à action immédiate est `RESTART_APP`. Ces Xid indiquent des erreurs au niveau de l'application pour lesquelles NVIDIA recommande de redémarrer l'application concernée plutôt que l'instance. Utilisez ces métriques pour détecter les pannes GPU transitoires affectant une tâche ou un conteneur spécifique sans impliquer une défaillance matérielle sous-jacente.

Les codes Xid suivants sont inclus. Pour la liste complète et les descriptions, consultez Erreurs NVIDIA Xid documentation.

Xid	Error type (Type d'erreur)
8	Le processeur graphique a arrêté le traitement
11	Flux de mémoire Push Buffer non valide ou endommagé
13	Exception relative au moteur graphique

Xid	Error type (Type d'erreur)
25	Suppression de la page ECC en cours
31	Défaut de page mémoire du GPU
32	Flux de mémoire Push Buffer non valide ou endommagé
39	Erreur de bus
40	Exception relative au processeur vidéo
41	Défaillance inattendue
60	Exception relative au processeur vidéo
68	Exception relative au processeur vidéo
69	Erreur de classe du moteur graphique
70	Erreur de canal utilisateur CE
71	Délai d'expiration du sémaphore du GPU
72	Erreur d'accès au sémaphore du GPU
75	Événement sur la liste noire de la page d'information
76	Erreur du moteur d'affichage
77	Erreur du moteur d'affichage
80	Données corrompues envoyées au GPU
82	Erreur NVJPG
83	Erreur NVDEC
84	Lien SLI non apparié
85	Contrainte de ressources

Xid	Error type (Type d'erreur)
86	Erreur du système d'exploitation
88	Erreur NVDEC
89	Erreur NVENC
94	Erreur ECC contenue
96	Erreur NVDEC
97	Erreur NVDEC
98	Erreur NVDEC
99	Erreur NVJPG
100	Erreur NVJPG
101	Erreur NVJPG
102	Erreur NVJPG
103	Délai d'expiration GSP RPC
104	Arrêt du GPS
105	Erreur GSP
126	Erreur de relecture C2C NVLink
127	Erreur C2C NVLink
128	Erreur NVLink
129	Erreur de restauration NVLink
130	Erreur fatale NVLink
131	Erreur non fatale NVLink

Xid	Error type (Type d'erreur)
132	Erreur NVLink
133	Erreur NVLink
134	Erreur NVLink
135	Erreur NVLink
139	Événement d'enregistrement de retrait d'une page de mémoire GPU

Surveillance des instances gérées Amazon ECS

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances de vos charges de travail Amazon ECS Managed Instances. AWS fournit plusieurs outils et services pour vous aider à surveiller vos applications et votre infrastructure conteneurisées.

Surveillance Container Insights

CloudWatch Container Insights fournit une surveillance complète de vos applications conteneurisées et de vos microservices. Container Insights collecte, agrège et résume automatiquement les métriques et les journaux de vos applications conteneurisées et microservices exécutés sur les instances gérées Amazon ECS.

Container Insights collecte des informations au niveau du cluster, du service et des tâches, offrant ainsi une visibilité sur :

- Utilisation de l’UC et de la mémoire
- Métriques des performances réseau
- Utilisation du stockage
- Performance des tâches et des services

Les statistiques sont disponibles dans CloudWatch tableaux de bord et peuvent être utilisés pour créer des alarmes et des réponses automatisées aux problèmes de performances. Container Insights fournit également des fonctionnalités de surveillance améliorées qui vous aident à identifier et à résoudre les problèmes rapidement.

> **Note**
>
> Container Information est disponible moyennant un coût supplémentaire. Pour plus d'informations sur la tarification, consultez Tarification d'CloudWatch .

Surveillance des GPU

Pour les instances gérées Amazon ECS exécutant NVIDIA GPU-enabled Types d'instances Amazon EC2, Container Insights, doté d'une observabilité améliorée, collecte des métriques GPU à partir de NVIDIA Data Center GPU Manager (DCGM) au niveau du conteneur, de la tâche et de l'instance. Les métriques GPU ne sont pas collectées avec Container Insights de base ; activez une meilleure observabilité pour accéder à la télémétrie GPU.

Aucune installation d'agent supplémentaire n'est requise. Les métriques GPU sont collectées automatiquement sur les types d'instances pris en charge lorsque Container Insights avec une observabilité améliorée est activé sur le cluster.

Pour obtenir la liste des métriques et des dimensions du GPU, voir Amazon ECS Container Insights avec des mesures d'observabilité améliorées.

Surveillance des instances

Pour surveiller l'infrastructure sous-jacente qui prend en charge vos charges de travail Amazon ECS Managed Instances, vous pouvez utiliser les métriques Amazon EC2 disponibles via CloudWatch.

Les instances gérées Amazon ECS sont fournies avec deux volumes Amazon EBS :

- Volume racine utilisé pour le système de fichiers du système d'exploitation
- Volume de données utilisé par les applications

Lorsque Container Insights est activé, Amazon ECS publie automatiquement des mesures d'utilisation du système d'exploitation et du système de fichiers de données au niveau de l'instance.

Les mesures disponibles pour les instances gérées Amazon ECS incluent :

- Métriques Amazon EC2 : utilisation du processeur, performances du réseau, fonctionnement des disques et contrôles d'état. Pour plus d'informations, voir Surveillez vos instances à l'aide de CloudWatch

- Statistiques Amazon ECS (lorsque Container Insights est activé) : utilisation du système d'exploitation et du système de fichiers contenant des volumes de données. Pour plus d'informations, voir Statistiques Amazon ECS Container Insights
- Métriques Amazon EBS : IOPS, débit, latence de lecture et d'écriture. Pour plus d'informations, voir Amazon EBS CloudWatch indicateurs

Ces métriques sont automatiquement disponibles sans installation manuelle d'agent.

Surveillance détaillée des instances gérées Amazon ECS

CloudWatch propose deux catégories de surveillance : surveillance de base et surveillance détaillée. Par défaut, la surveillance basique est configurée pour votre instance gérée. Vous pouvez éventuellement activer une surveillance détaillée pour vous aider à identifier plus rapidement les problèmes opérationnels et à y remédier. Vous pouvez activer ou désactiver la surveillance détaillée lorsque vous créez ou mettez à jour un fournisseur de capacité d'instances gérées Amazon ECS.

L'activation de la surveillance détaillée d'une instance gérée n'affecte pas la surveillance des volumes Amazon EBS qui lui sont attachés.

Le tableau suivant met en évidence les différences entre la surveillance de base et la surveillance détaillée de vos instances gérées.

Type de surveillance	Description	Frais
Surveillance de base	Les métriques de contrôle de statut sont disponibles par périodes d'une minute. Toutes les autres métriques sont disponibles par périodes de cinq minutes.	Aucuns frais.
Surveillance détaillée	Toutes les métriques, y compris les métriques de contrôle de statut, sont disponibles par périodes d'une minute. Pour obtenir le niveau de données, vous devez l'activer spécifiqu ement pour l'instance gérée. Pour les instances gérées où vous avez activé la surveillance détaillée, vous pouvez également obtenir les données agrégées à partir de groupes d'instanc es gérées similaires.	Vous êtes facturé par métrique qu'Amazon ECS Managed Instances envoie à CloudWatch. Vous n'êtes pas facturé pour le stockage des données. Pour plus d'informations,

Type de surveillance	Description	Frais
		consultez Niveau payant et Exemple 1 – Surveillance détaillée EC2 sur la page de tarification CloudWatch .

Autorisations requises

Pour activer la surveillance détaillée d'une instance gérée, votre utilisateur doit être autorisé à utiliser l'action d'API `MonitorInstances`. Pour désactiver la surveillance détaillée d'une instance gérée, votre utilisateur doit être autorisé à utiliser l'action d'API `UnmonitorInstances`.

Détermination de l'état des tâches Amazon ECS à l'aide de la surveillance de l'état des conteneurs

Lorsque vous créez une définition de tâche, vous pouvez configurer une surveillance de l'état pour vos conteneurs. Les surveillances de l'état du conteneur sont des commandes qui s'exécutent localement sur un conteneur et valident l'état et la disponibilité des applications.

L'agent de conteneur Amazon ECS assure une surveillance et génère des rapports uniquement pour les vérifications d'état spécifiées dans la définition de la tâche. Amazon ECS n'assure pas la surveillance de l'état Docker qui est intégrée dans une image de conteneur, mais qui n'est pas spécifiée dans la définition du conteneur. Les paramètres de surveillance de l'état spécifiés dans une définition du conteneur remplacent toutes les surveillances de l'état Docker présentes dans l'image de conteneur.

Lorsqu'une surveillance de l'état est spécifiée dans une définition de tâche, le conteneur exécute le processus de surveillance à l'intérieur du conteneur, puis évalue le code de sortie afin de déterminer l'état de l'application.

La surveillance de l'état comprend les paramètres suivants :

- Commande : commande que le conteneur exécute pour déterminer si celle-ci est saine. Le tableau de chaînes peut commencer par `CMD` pour exécuter directement les arguments de la commande, ou par `CMD-SHELL` pour exécuter la commande avec le shell par défaut du conteneur.

Utilisez `CMD-SHELL` lorsque vous avez besoin de fonctionnalités du shell telles que les canaux, les redirections, le chaînage de commandes ou l'extension de variables d'environnement. Par exemple, `CMD-SHELL` vous permet d'utiliser des commandes telles que `curl -f http://localhost/ || exit 1` où le shell interprète l'opérateur `||`. Utilisez `CMD` pour les commandes simples qui ne nécessitent pas d'interprétation du shell.

- Intervalle : durée (en secondes) entre chaque surveillance de l'état.
- Délai d'expiration : durée (en secondes) pendant laquelle le système attend que la surveillance de l'état aboutisse avant de la considérer comme un échec.
- Réessais : nombre de fois où une surveillance de l'état échoué doit être réessayée avant que le conteneur ne soit considéré comme défectueux.
- Période de démarrage : période de grâce facultative accordée aux conteneurs pour leur permettre de démarrer avant que les échecs de la surveillance de l'état ne soient pris en compte dans le nombre maximal de tentatives.

  Si une vérification de l'état aboutit dans le délai défini par `startPeriod`, le conteneur est considéré comme sain et toutes les défaillances suivantes sont prises en compte dans le nombre maximal de nouvelles tentatives.

Pour plus d'informations sur la manière de spécifier une surveillance de l'état dans la définition d'une tâche, consultez la section Surveillance de l'état.

Voici une description des valeurs d'état possibles pour un conteneur :

- `HEALTHY` : le conteneur a passé avec succès la surveillance de l'état.
- `UNHEALTHY` : le conteneur a échoué à la surveillance de l'état.
- `UNKNOWN` : la surveillance de l'état du conteneur est en cours d'évaluation, aucune surveillance de l'état du conteneur n'est définie ou Amazon ECS n'a pas l'état de santé du conteneur.

Les commandes de surveillance de l'état s'exécutent sur le conteneur. Vous devez donc inclure les commandes dans l'image du conteneur.

La surveillance de l'état se connecte à l'application via l'interface de bouclage du conteneur à `localhost` ou `127.0.0.1`. Un code de sortie `0` indique la réussite et tout code de sortie autre que zéro indique qu'il s'agit d'un échec.

Tenez compte des éléments suivants lorsque vous utilisez la surveillance de l'état des conteneurs :

- Les surveillances de l'état du conteneur requièrent la version 1.17.0 ou une version supérieure de l'agent de conteneur Amazon ECS.
- Les surveillances de l’état du conteneur sont prises en charge pour les tâches Fargate si vous utilisez une version de plateforme Linux `1.1.0` ou ultérieure ou une version de plateforme Windows `1.1.0` ou ultérieure

Comment Amazon ECS détermine l’état des tâches

Seuls les conteneurs essentiels et disposant d’une commande de surveillance de l’état dans la définition de la tâche sont pris en compte pour déterminer l’état de la tâche.

Les règles suivantes sont évaluées dans l’ordre :

1. Si le statut d’un conteneur essentiel est `UNHEALTHY`, le statut de la tâche est `UNHEALTHY`.
2. Si le statut d’un conteneur essentiel est `UNKNOWN`, le statut de la tâche est `UNKNOWN`.
3. Si le statut de tous les conteneurs essentiels est `HEALTHY`, le statut de la tâche est `HEALTHY`.

Prenons l’exemple d’état des tâches suivant avec deux conteneurs essentiels.

État du conteneur 1	État du conteneur 2	État de la tâche
UNHEALTHY	UNKNOWN	UNHEALTHY
UNHEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY

Prenons l’exemple d’état de la tâche suivant avec trois conteneurs essentiels.

État du conteneur 1	État du conteneur 2	État du conteneur 3	État de la tâche
UNHEALTHY	UNKNOWN	UNKNOWN	UNHEALTHY
UNHEALTHY	UNKNOWN	HEALTHY	UNHEALTHY

État du conteneur 1	État du conteneur 2	État du conteneur 3	État de la tâche
UNHEALTHY	HEALTHY	HEALTHY	UNHEALTHY
HEALTHY	UNKNOWN	HEALTHY	UNKNOWN
HEALTHY	UNKNOWN	UNKNOWN	UNKNOWN
HEALTHY	HEALTHY	HEALTHY	HEALTHY

Comment les déconnexions des agents affectent la surveillance de l'état

Si l'agent de conteneur Amazon ECS se déconnecte du service Amazon ECS, cela n'entraînera pas le passage du conteneur à l'état `UNHEALTHY`. Ce mécanisme a été conçu pour garantir que les conteneurs continuent de fonctionner pendant les redémarrages de l'agent ou en cas d'indisponibilité temporaire. Le statut de la surveillance de l'état correspond à la « dernière réponse reçue » de l'agent Amazon ECS. Ainsi, si le conteneur était considéré comme `HEALTHY` avant la déconnexion, ce statut restera inchangé jusqu'à ce que l'agent se reconnecte et qu'un autre contrôle d'intégrité soit effectué. Aucune hypothèse n'a été émise quant au statut des surveillances de l'état des conteneurs.

Affichage de l'état d'un conteneur Amazon ECS

Vous pouvez consulter l'état du conteneur dans la console et utiliser l'API dans la réponse `DescribeTasks`. Pour plus d'informations, voir DescribeTasks dans le Référence de l'API Amazon Elastic Container Service.

Si vous utilisez la journalisation pour votre conteneur, par exemple Amazon CloudWatch Journaux, vous pouvez configurer la commande de vérification de l'état de santé pour transmettre le résultat de l'état du conteneur à vos journaux. Assurez-vous de l'utiliser `2&1` pour attraper à la fois les informations `stdout` et `stderr`. L'exemple suivant l'utilise `CMD-SHELL`, car il nécessite des fonctionnalités du shell telles que les opérateurs pipe (>>) et logique OR (||) :

```
"command": [
     "CMD-SHELL",
     "curl -f http://localhost/ >> /proc/1/fd/1 2>&1  || exit 1"
   ],
```

Surveillance de l’état des instances de conteneurs Amazon ECS

Amazon ECS fournit la surveillance de l'état des instances de conteneur. Vous pouvez rapidement déterminer si Amazon ECS a détecté des problèmes susceptibles d'empêcher vos instances de conteneur d'exécuter des conteneurs. Amazon ECS effectue des contrôles automatisés sur chaque instance de conteneur en cours d'exécution avec la version d'agent `1.57.0` ou ultérieure pour identifier les problèmes. Pour plus d'informations sur la vérification de la version d'agent et d'une instance de conteneur, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Vous devez utiliser AWS CLI version `1.22.3` ou plus tard ou AWS CLI version `2.3.6` ou plus tard. Pour plus d'informations sur la mise à jour du AWS CLI, voir Installation ou mise à jour de la dernière version du AWS CLI dans le AWS Command Line Interface Guide de l'utilisateur version 2.

Pour afficher l’état de l’instance de conteneur, exécutez`describe-container-instances` avec l’option `CONTAINER_INSTANCE_HEALTH`.

Le `overallStatus` est déterminé par les états des bilans de santé individuels dans le `details` tableau. L'état le plus grave est prioritaire dans l'ordre suivant : `IMPAIRED`, `INSUFFICIENT_DATA`, `INITIALIZING`, et `OK`.

Les valeurs suivantes sont valides pour `overallStatus` :

- `OK` — Tous les bilans de santé sont réussis.
- `IMPAIRED` — Un ou plusieurs bilans de santé ont échoué.
- `INSUFFICIENT_DATA` — Les données relatives à un ou plusieurs bilans de santé ne sont pas disponibles.
- `INITIALIZING` — Un ou plusieurs bilans de santé sont en cours d'initialisation.

Le bilan de santé `type` les valeurs incluent :

- `CONTAINER_RUNTIME` — Surveille l'état de santé du moteur d'exécution du conteneur (par exemple, le démon Docker). Disponible sur le type de lancement EC2.
- `ACCELERATED_COMPUTE` — Surveille l'état des périphériques de calcul accéléré tels que les GPU. Lorsqu'un GPU est endommagé, `statusReason` contient le code d'erreur NVIDIA Xid au format `XID_<number>`. Disponible sur le type de lancement Amazon ECS Managed Instances.
- `DAEMON` — Surveille l'état des tâches de démon requises sur l'instance de conteneur. Disponible sur le type de lancement Amazon ECS Managed Instances.

- `AGENT_CONNECTIVITY` — Surveille la connectivité entre l'agent de conteneur Amazon ECS et le plan de contrôle Amazon ECS. Disponible sur les instances gérées Amazon ECS, AWS Fargate, et les types de lancement d'Amazon EC2.

Vous pouvez également suivre l'évolution de l'état de santé par le biais d'événements. Pour plus d'informations, consultez Événements de modification de l'état de santé des instances de conteneurs Amazon ECS.

Voici un exemple qui montre comment exécuter `describe-container-instances` :

```
aws ecs describe-container-instances \
     --cluster cluster_name \
     --container-instances 47279cd2cadb41cbaef2dcEXAMPLE \
     --include CONTAINER_INSTANCE_HEALTH
```

Voici un exemple de l'objet du statut de l'état dans la sortie.

```
"healthStatus": {
 "overallStatus": "OK",
 "details": [{
  "type": "CONTAINER_RUNTIME",
  "status": "OK",
  "lastUpdated": "2021-11-10T03:30:26+00:00",
  "lastStatusChange": "2021-11-10T03:26:41+00:00"
 }]
}
```

Problèmes liés à l'état des instances de conteneur

Si le statut `overallStatus` est autre que `OK`, essayez ce qui suit :

- Attendez, puis exécutez `describe-container-instances`
- Affichez l'état de votre instance de conteneur dans la console EC2 ou à l'aide de la CLI.
- Passez en revue le CloudWatch métriques. Pour plus d'informations, consultez Surveillez Amazon ECS à l'aide CloudWatch
- Vérifiez le Tableau de bord AWS Health pour voir s'il y a des problèmes avec le service.

Identification des opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications

Amazon ECS s'intègre à AWS Distro pour OpenTelemetry pour collecter des données de trace à partir de votre application. Amazon ECS utilise un AWS Distro pour OpenTelemetry conteneur sidecar pour collecter et acheminer les données de suivi vers AWS X-Ray. Pour plus d'informations, voir Configuration AWS Distro pour OpenTelemetry Collector dans Amazon ECS. Vous pouvez ensuite utiliser AWS X-Ray pour identifier les erreurs et les exceptions, analyser les goulots d'étranglement en matière de performances et les temps de réponse.

Pour le AWS Distro pour OpenTelemetry Collecteur auquel envoyer les données de trace AWS X-Ray, votre application doit être configurée pour créer les données de trace. Pour plus d'informations, consultez Instrumenter votre application pour AWS X-Ray dans le Guide du développeur AWS X-Ray .

Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec AWS X-Ray

L'intégration d'Amazon ECS avec AWS Distro pour OpenTelemetry nécessite que vous créiez un rôle de tâche et que vous le spécifiiez dans votre définition de tâche. Nous vous recommandons de configurer le AWS Distro pour OpenTelemetry sidecar pour acheminer les grumes des conteneurs vers CloudWatch Journaux.

> **Important**
>
> Si vous collectez également des statistiques d'application à l'aide du AWS Distro pour OpenTelemetry intégration, assurez-vous que votre rôle IAM de tâche contient également les autorisations nécessaires à cette intégration. Pour plus d'informations, consultez Corrélation des performances des applications Amazon ECS à l'aide des métriques d'application.

Après avoir créé le rôle, créez une politique avec les autorisations suivantes, puis associez-la au rôle.

- `logs:PutLogEvents`
- `logs:CreateLogGroup`
- `logs:CreateLogStream`
- `logs:DescribeLogStreams`

- logs:DescribeLogGroups
- logs:PutRetentionPolicy
- xray:PutTraceSegments
- xray:PutTelemetryRecords
- xray:GetSamplingRules
- xray:GetSamplingTargets
- xray:GetSamplingStatisticSummaries
- ssm:GetParameters

Spécifier le AWS Distro pour OpenTelemetry sidecar pour AWS X-Ray intégration dans votre définition de tâches

La console Amazon ECS simplifie la création de AWS Distro pour OpenTelemetry conteneur sidecar en utilisant le Utiliser la collecte de traces option. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter le AWS Distro pour OpenTelemetry conteneur sidecar à la définition de votre tâche. L'extrait de définition de tâche suivant montre la définition du conteneur pour ajouter le AWS Distro pour OpenTelemetry sidecar pour AWS X-Ray intégration.

```
{
 "family": "otel-using-xray",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryXrayRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   },
```

```
    "dependsOn": [{
     "containerName": "aws-otel-collector",
     "condition": "START"
    }]
   },
   {
    "name": "aws-otel-collector",
    "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
    "essential": true,
    "command": [
     "--config=/etc/ecs/otel-instance-metrics-config.yaml"
    ],
    "logConfiguration": {
     "logDriver": "awslogs",
     "options": {
      "awslogs-create-group": "True",
      "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
      "awslogs-region": "us-east-1",
      "awslogs-stream-prefix": "ecs"
     }
    }
   }
  ],
  "networkMode": "awsvpc",
  "requiresCompatibilities": [
   "FARGATE"
  ],
  "cpu": "1024",
  "memory": "3072"
}
```

Corrélation des performances des applications Amazon ECS à l'aide des métriques d'application

Amazon ECS sur Fargate permet de collecter des métriques à partir de vos applications exécutées sur Fargate et de les exporter vers Amazon CloudWatch ou Amazon Managed Service pour Prometheus.

Vous pouvez utiliser les métadonnées collectées pour corréler les données de performance des applications avec les données d'infrastructure sous-jacentes, réduisant ainsi le temps moyen de résolution du problème.

Amazon ECS utilise un AWS Distro pour OpenTelemetry conteneur sidecar pour collecter et acheminer les métriques de votre application vers la destination. L'expérience de la console Amazon ECS simplifie le processus d'ajout de cette intégration lors de la création de vos définitions de tâches.

Rubriques

- Exporter les métriques des applications vers Amazon CloudWatch
- Exportation des métriques d'application vers Amazon Managed Service for Prometheus

Exporter les métriques des applications vers Amazon CloudWatch

Amazon ECS sur Fargate permet d'exporter les métriques personnalisées de votre application vers Amazon CloudWatch sous forme de métriques personnalisées. Cela se fait en ajoutant le AWS Distro pour OpenTelemetry conteneur sidecar à la définition de votre tâche. La console Amazon ECS simplifie ce processus en ajoutant l'option Utiliser la collection de métriques lors de la création d'une définition de tâche. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Les métriques de l'application sont exportées vers CloudWatch Journaux avec nom de groupe de journaux `/aws/ecs/application/metrics` et les statistiques peuvent être consultées dans le `ECS/AWSOTel/Application` espace de noms. Votre candidature doit être instrumentée avec le OpenTelemetry SDK. Pour plus d'informations, voir Présentation de AWS Distro pour OpenTelemetry dans le AWS Distro pour OpenTelemetry documentation.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de l'intégration d'Amazon ECS sur Fargate avec AWS Distro pour OpenTelemetry pour envoyer les statistiques de l'application à Amazon CloudWatch.

- Cette intégration envoie uniquement les métriques personnalisées de votre application à CloudWatch. Si vous souhaitez obtenir des statistiques au niveau des tâches, vous pouvez activer Container Insights dans la configuration du cluster Amazon ECS. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.
- Le AWS Distro pour OpenTelemetry l'intégration est prise en charge pour les charges de travail Amazon ECS hébergées sur Fargate et les charges de travail Amazon ECS hébergées sur des instances Amazon EC2. Les instances externes ne sont actuellement pas prises en charge.

- CloudWatch prend en charge un maximum de 30 dimensions par métrique. Par défaut, Amazon ECS inclut par défaut les dimensions `TaskARN`, `ClusterARN`, `LaunchType`, `TaskDefinitionFamily`, et `TaskDefinitionRevision` aux métriques. Les 25 dimensions restantes peuvent être définies par votre application. Si plus de 30 dimensions sont configurées, CloudWatch Impossible de les afficher. Dans ce cas, les statistiques de l'application apparaîtront dans le `ECS/AWSOTel/Application` CloudWatch espace de noms métrique mais sans aucune dimension. Vous pouvez instrumenter votre application pour ajouter des dimensions supplémentaires. Pour plus d'informations, voir En utilisant CloudWatch métriques avec AWS Distro pour OpenTelemetry dans le AWS Distro pour OpenTelemetry documentation.

Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec Amazon CloudWatch

L'intégration d'Amazon ECS avec AWS Distro pour OpenTelemetry nécessite que vous créiez un rôle IAM pour les tâches et que vous spécifiiez ce rôle dans votre définition de tâche. Nous recommandons que AWS Distro pour OpenTelemetry le sidecar doit également être configuré pour acheminer les journaux des conteneurs vers CloudWatch Les journaux qui nécessitent un rôle IAM pour l'exécution des tâches doivent être créés et spécifiés également dans votre définition de tâche. La console Amazon ECS prend en charge le rôle IAM d'exécution de tâche pour vous, mais le rôle IAM de tâche doit être créé manuellement et ajouté à votre définition de tâche. Pour plus d'informations sur le rôle IAM d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Si vous collectez également des données de suivi des applications à l'aide du AWS Distro pour OpenTelemetry intégration, assurez-vous que votre rôle IAM de tâche contient également les autorisations nécessaires à cette intégration. Pour plus d'informations, consultez Identification des opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications.
>
> Si votre application nécessite des autorisations supplémentaires, vous devez les ajouter à cette politique. Chaque définition de tâche ne peut spécifier qu'un rôle IAM de tâche. Par exemple, si vous utilisez un fichier de configuration personnalisé stocké dans Systems Manager, vous devez ajouter l'autorisation `ssm:GetParameters` à cette politique IAM.

Pour créer un rôle de service Elastic Container Service (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Rôles, puis Créer un rôle.
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation Tâche Elastic Container Service.
5. Choisissez Suivant.
6. Dans le Ajouter des autorisations section, recherchez AWSDistroOpenTelemetryPolicyForXray, puis sélectionnez la politique.
7. (Facultatif) Définissez une limite d'autorisations. Il s'agit d'une fonctionnalité avancée disponible pour les rôles de service, mais pas 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.

      L'IAM inclut une liste des AWS politiques gérées et gérées par le client sur votre compte.

   b. Sélectionnez la politique à utiliser comme limite d'autorisations.
8. Choisissez Suivant.
9. Saisissez un nom de rôle ou un suffixe de nom de rôle pour vous aider à identifier l'objectif du rôle.

   Important

   Lorsque vous nommez un rôle, notez ce qui suit :

   - Les noms de rôles doivent être uniques dans votre Compte AWS, et ne peut pas être rendu unique 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 Tags pour 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).

Spécifier le AWS Distro pour OpenTelemetry sidecar dans votre définition de tâche

La console Amazon ECS simplifie l'expérience de création du AWS Distro pour OpenTelemetry conteneur sidecar en utilisant le Utiliser la collecte de métriques option. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter AWS Distro pour OpenTelemetry conteneur sidecar à votre définition de tâche manuellement. L'exemple de définition de tâche suivant montre la définition du conteneur pour ajouter AWS Distro pour OpenTelemetry sidecar pour Amazon CloudWatch intégration.

```
{
 "family": "otel-using-cloudwatch",
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [
    {
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
```

```
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-cloudwatch.yaml"
   ],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "us-east-1",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Exportation des métriques d'application vers Amazon Managed Service for Prometheus

Amazon ECS prend en charge l'exportation de vos métriques de processeur, de mémoire, de réseau et de stockage au niveau des tâches, ainsi que vos métriques d'application personnalisées vers Amazon Managed Service for Prometheus. Cela se fait en ajoutant AWS Distro pour OpenTelemetry conteneur sidecar à la définition de votre tâche. La console Amazon ECS simplifie ce processus en ajoutant l'option Utiliser la collection de métriques lors de la création d'une définition de tâche. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Les métriques sont exportées vers Amazon Managed Service for Prometheus et peuvent être consultées à l'aide du tableau de bord Amazon Managed Grafana. Votre application doit être instrumentée soit avec les bibliothèques Prometheus, soit avec OpenTelemetry SDK. Pour plus d'informations sur l'instrumentation de votre application avec OpenTelemetry SDK, voir Présentation de AWS Distro pour OpenTelemetry dans le AWS Distro pour OpenTelemetry documentation.

Lorsque vous utilisez les bibliothèques Prometheus, votre application doit exposer un point de terminaison `/metrics` utilisé pour récupérer les données de métriques. Pour plus d'informations sur l'instrumentation de votre application avec les bibliothèques Prometheus, consultez Bibliothèques clientes Prometheus dans la documentation Prometheus.

Considérations

Les points suivants doivent être pris en compte lors de l'utilisation de l'intégration d'Amazon ECS sur Fargate avec AWS Distro pour OpenTelemetry pour envoyer les métriques de l'application à Amazon Managed Service for Prometheus.

- Le AWS Distro pour OpenTelemetry l'intégration est prise en charge pour les charges de travail Amazon ECS hébergées sur Fargate et les charges de travail Amazon ECS hébergées sur des instances Amazon EC2. Les instances externes ne sont actuellement pas prises en charge.
- Par défaut, AWS Distro pour OpenTelemetry inclut toutes les dimensions disponibles au niveau des tâches pour les métriques de votre application lors de l'exportation vers Amazon Managed Service pour Prometheus. Vous pouvez également instrumenter votre application pour ajouter des dimensions supplémentaires. Pour plus d'informations, voir Démarrez avec Prometheus Remote Write Exporter pour Amazon Managed Service for Prometheus dans le AWS Distro pour OpenTelemetry documentation.

Autorisations IAM requises pour AWS Distro pour OpenTelemetry intégration avec Amazon Managed Service pour Prometheus

L'intégration d'Amazon ECS à Amazon Managed Service pour Prometheus à l'aide du AWS Distro pour OpenTelemetry sidecar nécessite que vous créiez un rôle IAM de tâche et que vous spécifiiez ce rôle dans votre définition de tâche. Ce rôle IAM de tâche doit être créé manuellement avant d'enregistrer votre définition de tâche. Pour plus d'informations sur la création d'un rôle de tâche, consultez la section rôle IAM de tâche Amazon ECS.

Nous recommandons que le AWS Distro pour OpenTelemetry le sidecar doit également être configuré pour acheminer les journaux des conteneurs vers CloudWatch Les journaux qui nécessitent

un rôle IAM pour l'exécution des tâches doivent être créés et spécifiés également dans votre définition de tâche. La console Amazon ECS prend en charge le rôle IAM d'exécution de tâche pour vous, mais le rôle IAM de tâche doit être créé manuellement. Pour en savoir plus sur la création d'un rôle IAM d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Important**
>
> Si vous collectez également des données de suivi des applications à l'aide du AWS Distro pour OpenTelemetry intégration, assurez-vous que votre rôle IAM de tâche contient également les autorisations nécessaires à cette intégration. Pour plus d'informations, consultez Identification des opportunités d'optimisation d'Amazon ECS à l'aide des données de suivi des applications.

Les autorisations suivantes sont requises pour AWS Distro pour OpenTelemetry intégration avec Amazon Managed Service pour Prometheus :

- journaux :PutLogEvents
- journaux :CreateLogGroup
- journaux :CreateLogStream
- journaux :DescribeLogStreams
- journaux :DescribeLogGroups
- Cloudwatch :PutMetricData

Spécifier le AWS Distro pour OpenTelemetry sidecar dans votre définition de tâche

La console Amazon ECS simplifie l'expérience de création du AWS Distro pour OpenTelemetry conteneur sidecar en utilisant le Utiliser la collecte de métriques option. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.

Si vous n'utilisez pas la console Amazon ECS, vous pouvez ajouter le AWS Distro pour OpenTelemetry conteneur sidecar à votre définition de tâche manuellement. L'exemple de définition de tâche suivant montre la définition du conteneur pour ajouter AWS Distro pour OpenTelemetry sidecar pour l'intégration d'Amazon Managed Service pour Prometheus.

```
{
 "family": "otel-using-cloudwatch",
```

```
 "taskRoleArn": "arn:aws:iam::111122223333:role/AmazonECS_OpenTelemetryCloudWatchRole",
 "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole",
 "containerDefinitions": [{
   "name": "aws-otel-emitter",
   "image": "application-image",
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "true",
     "awslogs-group": "/ecs/aws-otel-emitter",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   },
   "dependsOn": [{
    "containerName": "aws-otel-collector",
    "condition": "START"
   }]
  },
  {
   "name": "aws-otel-collector",
   "image": "public.ecr.aws/aws-observability/aws-otel-collector:v0.30.0",
   "essential": true,
   "command": [
    "--config=/etc/ecs/ecs-amp.yaml"
   ],
   "environment": [{
    "name": "AWS_PROMETHEUS_ENDPOINT",
    "value": "https://aps-workspaces.aws-region.amazonaws.com/workspaces/
ws-a1b2c3d4-5678-90ab-cdef-EXAMPLE11111/api/v1/remote_write"
   }],
   "logConfiguration": {
    "logDriver": "awslogs",
    "options": {
     "awslogs-create-group": "True",
     "awslogs-group": "/ecs/ecs-aws-otel-sidecar-collector",
     "awslogs-region": "aws-region",
     "awslogs-stream-prefix": "ecs"
    }
   }
  }
 ],
 "networkMode": "awsvpc",
 "requiresCompatibilities": [
```

```
  "FARGATE"
 ],
 "cpu": "1024",
 "memory": "3072"
}
```

Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail

Amazon Elastic Container Service est intégré à AWS CloudTrail, un service qui fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou Service AWS. CloudTrail capture tous les appels d'API pour Amazon ECS sous forme d'événements. Les appels capturés comprennent les ceux provenant de la console Amazon ECS et les appels de code vers les opérations de l'API Amazon ECS. Utilisation des informations collectées par CloudTrail, vous pouvez déterminer la demande qui a été envoyée à Amazon ECS, l'adresse IP à partir de laquelle la demande a été faite, la date à laquelle elle a été faite et des informations supplémentaires.

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 :

- Si la demande a été effectuée avec des informations d'identification d'utilisateur root ou d'utilisateur root.
- Si la demande a été faite au nom d'un utilisateur du centre d'identité 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 requête a été effectuée par un autre Service AWS.

CloudTrail est actif dans votre Compte AWS lorsque vous créez le compte et que vous avez automatiquement accès au CloudTrail Historique de l'événement. Le CloudTrail Historique de l'événement fournit un enregistrement consultable, consultable, téléchargeable et immuable des 90 derniers jours des événements de gestion enregistrés dans un Région AWS. Pour plus d'informations, voir Travailler avec CloudTrail Historique de l'événement dans le AWS CloudTrail Guide de l'utilisateur. Il n'y a pas CloudTrail frais de visionnage du Historique de l'événement.

Pour un enregistrement continu des événements survenus dans votre Compte AWS au cours des 90 derniers jours, créez un parcours ou CloudTrail Lac magasin de données d'événements.

CloudTrail sentiers

UNE sentier permet CloudTrail pour transmettre des fichiers journaux à un compartiment Amazon S3. Tous les sentiers créés à l'aide du Console de gestion AWS sont multirégionaux. Vous ne pouvez créer un journal de suivi en une ou plusieurs régions à l'aide de l' AWS CLI. Il est recommandé de créer un sentier multirégional car vous capturez l'activité dans tous les domaines Régions AWS dans votre compte. Si vous créez un journal de suivi pour une seule région, il convient de n'afficher que les événements enregistrés dans le journal de suivi pour une seule région Région AWS. Pour plus d'informations sur les journaux de suivi, consultez Créez un journal de suivi dans vos Compte AWS et Création d'un journal de suivi pour une organisation dans le AWS CloudTrail Guide de l'utilisateur.

Vous pouvez envoyer gratuitement une copie de vos événements de gestion en cours dans votre compartiment Amazon S3 à CloudTrail en créant une trace, cependant, des frais de stockage Amazon S3 s'appliquent. Pour plus d'informations sur CloudTrail tarification, voir AWS CloudTrail Tarification. Pour obtenir des informations sur la tarification Amazon S3, consultez Tarification Amazon S3.

CloudTrail Magasins de données sur les événements liés

CloudTrail Lac vous permet de courir SQL-based des questions sur vos événements. CloudTrail Lake convertit les événements existants au format JSON basé sur des lignes en ORC Apache format. ORC est un format de stockage en colonnes qui est optimisé pour une récupération rapide des données. Les événements sont agrégés dans des magasins de données d'événement. Ceux-ci constituent des collections immuables d'événements basées sur des critères que vous sélectionnez en appliquant des sélecteurs d'événements avancés. Les sélecteurs que vous appliquez à un magasin de données d'événement contrôlent les événements qui persistent et que vous pouvez interroger. Pour plus d'informations sur CloudTrail Lac, voir Travailler avec AWS CloudTrail Lac dans le AWS CloudTrail Guide de l'utilisateur.

CloudTrail Les stockages et les requêtes de données sur les événements liés aux lacs entraînent des coûts. Lorsque vous créez un magasin de données d'événement, vous choisissez l'option de tarification que vous voulez utiliser pour le magasin de données d'événement. L'option de tarification détermine le coût d'ingestion et de stockage des événements, ainsi que les périodes de conservation par défaut et maximale pour le magasin de données d'événement. Pour plus d'informations sur CloudTrail tarification, voir AWS CloudTrail Tarification.

Événements de gestion Amazon ECS dans CloudTrail

Événements de gestion fournir des informations sur les opérations de gestion effectuées sur les ressources de votre Compte AWS. Ils sont également connus sous le nom opérations de plan de contrôle. Par défaut, CloudTrail enregistre les événements de gestion.

Amazon Elastic Container Service journalise toutes les opérations du plan de contrôle Amazon ECS en tant qu'événements de gestion. Par exemple, les appels au `CreateService`, `RunTask` et `DeleteCluster` les sections génèrent des entrées dans CloudTrail fichiers journaux. Pour obtenir la liste des opérations du plan de contrôle Amazon Elastic Container Service auxquelles Amazon ECS se connecte CloudTrail, consultez le Référence de l'API Amazon Elastic Container Service.

Événements liés aux données Amazon ECS dans CloudTrail

Événements liés aux données fournir des informations sur les opérations de ressources effectuées sur ou dans une ressource. Par défaut, CloudTrail n'enregistre pas les événements liés aux données. Le CloudTrail Historique de l'événement n'enregistre pas les événements liés aux données.

Des frais supplémentaires s'appliquent pour les événements de données. Pour plus d'informations sur CloudTrail tarification, voir AWS CloudTrail Tarification.

Vous pouvez enregistrer les événements de données pour les types de ressources Amazon ECS à l'aide du CloudTrail console, AWS CLI, ou CloudTrail Opérations d'API. Pour plus d'informations sur la façon de journaliser les événements de données, consultez Journalisation des événements de données avec la Console de gestion AWS et Journalisation des événements de données avec l' AWS Command Line Interface dans le Guide de l'utilisateur AWS CloudTrail .

Le tableau suivant répertorie les types de ressources Amazon ECS pour lesquels vous pouvez enregistrer les événements liés aux données. Le Type d'événement de données (console) la colonne indique la valeur à choisir parmi Type d'événement de données liste sur le CloudTrail console. Le valeur resources.type la colonne affiche `resources.type` valeur, que vous devez spécifier lors de la configuration de sélecteurs d'événements avancés à l'aide du AWS CLI ou CloudTrail API. Le API de données connectées à CloudTrail la colonne affiche les appels d'API enregistrés dans CloudTrail pour le type de ressource.

Type d’événement de données (console)	valeur resources.type	API de données connectées à CloudTrail
AwsApiCall	`AWS::ECS::ContainerInstance`	• `ecs:Poll` (pour les instances gérées EC2 et Amazon ECS) • `ecs:StartTelemetrySession` (pour les instances gérées EC2 et Amazon ECS) • `ecs:PutSystemLogEvents` (pour les instances gérées Amazon ECS)

Vous pouvez configurer des sélecteurs d’événements avancés pour filtrer les champs `eventName`, `readOnly` et `resources.ARN` afin de ne journaliser que les événements importants pour vous. Pour plus d’informations sur ces champs, consultez AdvancedFieldSelector dans la Référence des API AWS CloudTrail .

Exemples d’événements Amazon ECS

Un événement représente une demande unique provenant de n'importe quelle source et inclut des informations sur l'opération d'API demandée, la date et l'heure de l'opération, les paramètres de la demande, etc. CloudTrail les fichiers journaux ne constituent pas une pile ordonnée des appels d'API publics, de sorte que les événements n'apparaissent pas dans un ordre spécifique.

L'exemple suivant montre un CloudTrail événement qui démontre `CreateService` action.

> **Note**
>
> Cet exemple a été mis en forme pour faciliter sa lecture. Dans un CloudTrail fichier journal, toutes les entrées et tous les événements sont concaténés sur une seule ligne. En outre, cet exemple se limite à une seule entrée Amazon ECS. Dans un vrai CloudTrail fichier journal, vous voyez les entrées et les événements de plusieurs AWS services.

```
{
    "eventVersion": "1.04",
    "userIdentity": {
        "type": "AssumedRole",
        "principalId": "AIDACKCEVSQ6C2EXAMPLE:account_name",
        "arn": "arn:aws:sts::123456789012:user/Mary_Major",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIOSFODNN7EXAMPLE",
        "sessionContext": {
            "attributes": {
                "mfaAuthenticated": "false",
                "creationDate": "2018-06-20T18:32:25Z"
            },
            "sessionIssuer": {
                "type": "Role",
                "principalId": "AIDACKCEVSQ6C2EXAMPLE",
                "arn": "arn:aws:iam::123456789012:role/Admin",
                "accountId": "123456789012",
                "userName": "Mary_Major"
            }
        }
    },
    "eventTime": "2018-06-20T19:04:36Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "CreateCluster",
    "awsRegion": "us-east-1",
    "sourceIPAddress": "203.0.113.12",
    "userAgent": "console.amazonaws.com",
    "requestParameters": {
        "clusterName": "default"
    },
    "responseElements": {
        "cluster": {
            "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/default",
            "pendingTasksCount": 0,
            "registeredContainerInstancesCount": 0,
            "status": "ACTIVE",
            "runningTasksCount": 0,
            "statistics": [],
            "clusterName": "default",
            "activeServicesCount": 0
        }
    },
```

```
    "requestID": "cb8c167e-EXAMPLE",
    "eventID": "e3c6f4ce-EXAMPLE",
    "eventType": "AwsApiCall",
    "recipientAccountId": "123456789012"
}
```

Pour plus d'informations sur CloudTrail contenu de l'enregistrement, voir CloudTrail contenu de l'enregistrement dans le AWS CloudTrail Guide de l'utilisateur.

Visualisation CloudWatch Logs Live Tail pour les services et tâches Amazon ECS

CloudWatch Logs Live Tail vous aide à résoudre rapidement les incidents en affichant une liste en streaming des nouveaux événements de journal au fur et à mesure de leur ingestion. Vous pouvez afficher les événements de tâche et de service dans la console Amazon ECS. Cette vue fournit une vue cohérente de l'état de vos tâches et de vos services.

Chaque tâche d'un service Amazon ECS possède des flux de journaux dédiés pour chaque conteneur. Si un service augmente progressivement, chaque instance de tâche possède son propre ensemble de flux de journaux. La convention de dénomination pour les flux de journaux suit le modèle suivant :

```
log-group-prefix/container-name/task-id
```

Une seule tâche écrit dans les mêmes flux de journalisation pendant toute sa durée de vie. Le flux de journaux contient les messages provenant des conteneurs de cette tâche, ainsi que toute sortie du code de votre application. Chaque message est horodaté, y compris vos journaux personnalisés.

> **Note**
>
> Le coût des sessions Live Tail est calculé en fonction du temps d'utilisation de la session, par minute. Pour plus d'informations sur la tarification, consultez Amazon CloudWatch Tarification.

Autorisations requises

Les autorisations suivantes sont requises pour que l'utilisateur IAM de la console puisse démarrer et arrêter CloudWatch Enregistre les sessions Live Tail :

- `logs:DescribeLogGroups`
- `logs:StartLiveTail`
- `logs:StopLiveTail`

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource que vous souhaitez afficher

Ressource	Étapes
Tâches	a. Sur la page Clusters, choisissez le cluster. La page des détails du cluster s’ouvre. b. Choisissez l'onglet Tasks. La page des détails de la tâche s’affiche. c. Sélectionnez l'onglet Logs (Journaux).
Services	a. Sur la page Clusters, choisissez le cluster. La page des détails du cluster s’ouvre. b. Sélectionnez le service. La page des détails du service s’affiche.

Ressource	Étapes
	c. Sélectionnez l'onglet Logs (Journaux).

3. Choisissez CloudWatch Journaux Live Tail, puis choisissez Démarrer.
4. (Facultatif) Pour filtrer les flux, sous Filtrer, pour Sélectionner les groupes de journaux, choisissez le groupe de journaux.

AWS CLI references

Vous pouvez également utiliser le AWS CLI pour démarrer des sessions Live Tail pour CloudWatch Journaux. Les ressources suivantes fournissent des informations détaillées sur l'utilisation du AWS CLI avec Live Tail :

- Référence de commande start-live-tail : syntaxe complète de commande, paramètres et exemples de commande `aws logs start-live-tail`.
- CloudWatch Guide de l'utilisateur de Logs Live Tail - Guide complet comprenant AWS CLI utilisation en mode impression uniquement et en mode interactif.
- StartLiveTail Exemples de SDK - Exemples programmatiques d'utilisation du StartLiveTail API avec différents AWS SDK.

Surveillez les opérations d'Amazon ECS avec Action Logs

Présentation d'Amazon ECS Action Logs

Avec Amazon ECS Action Logs, vous pouvez surveiller toutes les actions qu'Amazon ECS effectue en votre nom pendant les opérations liées au cycle de vie des conteneurs. Les journaux d'action capturent des enregistrements horodatés de chaque opération de ressource, de chaque transition d'état et de chaque appel d'API initié par un service au sein de vos clusters.

Les journaux d'actions enregistrent des opérations auparavant invisibles, telles que les téléchargements d'images de conteneurs, les enregistrements d'équilibreurs de charge et les configurations de groupes de sécurité. Sans journaux d'action, vous ne pouvez observer que les états de début et de fin de vos ressources. Grâce aux journaux d'action, vous pouvez consulter toutes les étapes intermédiaires franchies par Amazon ECS entre ces états.

Lorsque vous activez les journaux d'action, Amazon Q dans le AWS La console de gestion peut accéder à ces journaux pour vous aider à résoudre les échecs de déploiement et les problèmes liés aux démons directement depuis la console.

Avantages des journaux d'action

Les journaux d'action offrent les avantages suivants pour la gestion de vos charges de travail Amazon ECS :

- Visibilité des actions de service — Visualisez les actions auparavant invisibles qu'Amazon ECS effectue entre les transitions d'état, notamment le provisionnement de l'infrastructure, la configuration réseau et l'enregistrement des ressources.
- Résolution des problèmes améliorée — Diagnostiquez les échecs de déploiement et les problèmes d'infrastructure sans contacter AWS Soutien. Les journaux d'action prennent en charge un nombre croissant de raisons d'échec.
- Métadonnées de défaillance étendues — Conservez les métadonnées relatives aux tâches ayant échoué au-delà de la période de conservation standard d'une heure. Vous pouvez consulter les informations détaillées sur les défaillances à votre propre rythme.
- Self-service débogage — Étudiez et résolvez les problèmes de manière indépendante en passant en revue la séquence complète des actions qui ont conduit à une défaillance.

Ce qui est enregistré

Les journaux d'actions capturent les actions pour les opérations Amazon ECS suivantes :

- Déploiements de services — Transitions d'état, annulations et exécution de hooks de cycle de vie.
- Cycle de vie des démons gérés — Opérations de création, de mise à jour, de suppression et de vidange d'instance pour Managed Daemons.

Comment fonctionnent les journaux d'action

Vous pouvez accéder aux journaux d'actions au niveau du cluster. Vous pouvez activer les journaux d'action via la console Amazon ECS ou en utilisant Amazon CloudWatch API (`PutDeliverySource`, `PutDeliveryDestination`, et `CreateDelivery`).

Une fois que vous avez activé les journaux d'action, Amazon ECS émet des journaux JSON structurés via CloudWatch Centre d'ingestion. Amazon ECS livre les journaux à la destination que vous avez choisie : CloudWatch Logs, Amazon S3 ou Amazon Data Firehose.

Les journaux d'action utilisent la fonction de publication instantanée. La publication de journaux ne bloque ni ne dégrade jamais les opérations relatives aux ressources Amazon ECS. En cas d'échec de livraison temporaire, Amazon ECS continue de traiter vos charges de travail sans interruption.

Présentation du schéma de journalisation

Chaque entrée du journal des actions contient les champs suivants :

- `timestamp` — Millisecondes Unix lorsque l'événement s'est produit.
- `logLevel` — Le niveau de gravité : `INFO`, `WARN`, ou `ERROR`.
- `account` — Les AWS ID de compte.
- `region` — Les AWS Région où l'événement s'est produit.
- `resourceArn` — L'ARN du cluster qui représente la portée de l'opt-in.
- `actionSourceId` — Identifiant de sous-ressource, tel qu'un ARN de service ou un ARN de démon.
- `eventName` — L'identifiant de l'action, par exemple `DAEMON_DEPLOYMENT_IN_PROGRESS`.
- `detail` — Une charge utile JSON spécifique à l'événement avec un contexte supplémentaire sur l'action.

Destinations et organisation des journaux

Vous pouvez envoyer des journaux d'action vers les destinations suivantes :

- CloudWatch Journaux — CloudWatch Logs stocke les journaux dans un groupe de journaux nommé `/aws/vendedlogs/ecs/action-logs/{resourceId}`. Amazon ECS crée un flux de journaux distinct pour chaque sous-ressource, en utilisant `{actionSourceId}` comme nom du flux de journaux.
- Amazon S3 — Amazon S3 partitionne les journaux par compte, par région et par créneaux horaires de 5 minutes. Amazon S3 compresse les fichiers à l'aide de gzip pour un stockage efficace.
- Amazon Data Firehose — Amazon Data Firehose fournit des journaux sous forme d'enregistrements JSON délimités par de nouvelles lignes en temps quasi réel.

Tarification

Action Logs est une fonctionnalité payante qui utilise la norme CloudWatch tarification des grumes distribuées. Vous payez pour l'ingestion et le stockage des journaux en fonction du volume de journaux générés par vos clusters. Vous pouvez configurer la rétention via vos paramètres de destination. Lorsque vous activez les journaux d'action via la console Amazon ECS, CloudWatch Logs utilise une durée de conservation par défaut de 7 jours.

Rubriques

- Démarrez avec Amazon ECS Action Logs
- Résolution des problèmes liés aux journaux d'action Amazon ECS

Démarrez avec Amazon ECS Action Logs

Les journaux d'action Amazon ECS utilisent Amazon CloudWatch Mécanisme de distribution automatique des journaux pour publier des enregistrements détaillés des actions de service vers votre destination préférée. Lorsque vous activez les journaux d'action, vous pouvez voir chaque étape prise par Amazon ECS lors des déploiements et des opérations du démon, y compris le provisionnement de l'infrastructure, l'enregistrement des ressources et les détails des défaillances. Cette visibilité vous permet de résoudre les problèmes sans avoir à contacter AWS Soutenez et réduisez le temps nécessaire pour identifier les causes profondes. Vous pouvez activer les journaux d'action pour vos clusters à l'aide de la console Amazon ECS ou CloudWatch API de journalisation.

Conditions préalables

Avant d'activer les journaux d'action, vérifiez que votre identité IAM dispose des autorisations requises.

Autorisations IAM requises

Votre identité IAM doit disposer des autorisations suivantes pour activer et gérer les journaux d'actions :

```
{
    "Effect": "Allow",
    "Action": [
        "logs:PutDeliverySource",
        "logs:PutDeliveryDestination",
```

```
        "logs:CreateDelivery",
        "logs:GetDelivery",
        "ecs:AllowVendedLogDeliveryForResource"
    ],
    "Resource": "*"
}
```

Politique relative aux ressources de destination

La destination CloudWatch Le groupe de journaux Logs doit disposer d'une politique de ressources qui autorise `delivery.logs.amazonaws.com` service pour écrire des journaux. Ajoutez l'instruction suivante à la politique de ressources du groupe de journaux :

```
{
    "Effect": "Allow",
    "Principal": {
        "Service": "delivery.logs.amazonaws.com"
    },
    "Action": [
        "logs:CreateLogStream",
        "logs:PutLogEvents"
    ],
    "Resource": "arn:aws:logs:region:account-id:log-group:log-group-name:log-stream:*"
}
```

Activer les journaux d'action à l'aide de la console Amazon ECS

Vous pouvez activer les journaux d'action directement depuis la page des détails du cluster dans la console Amazon ECS.

Pour activer les journaux d'action à l'aide de la console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le volet de navigation, choisissez Clusters, puis sélectionnez le cluster.
3. Cliquez sur l'onglet Configuration.
4. Dans le Journaux d'actions section, choisissez Ajouter.
5. Pour Destination, choisissez l'une des options suivantes :

    - CloudWatch Journaux (par défaut)
    - Amazon S3

- Amazon Data Firehose

Pour CloudWatch Journaux, Amazon ECS crée un groupe de journaux par défaut nommé `/aws/vendedlogs/ecs/action-logs/`*`cluster-name`* avec une période de conservation de 7 jours.

6. Cliquez sur Enregistrer.

Activez les journaux d'action à l'aide CloudWatch API de journaux

Vous pouvez activer les journaux d'action par programmation à l'aide du CloudWatch Consigne les API de livraison de journaux distribuées. Cette approche nécessite trois étapes : enregistrer la source de diffusion, configurer la destination et créer la diffusion.

Pour activer les journaux d'actions à l'aide du AWS INTERFACE DE LIGNE DE COMMANDE (CLI)

1. Enregistrez la source de diffusion en exécutant la commande suivante :

```
aws logs put-delivery-source \
    --name my-ecs-action-logs \
    --resource-arn arn:aws:ecs:region:account-id:cluster/cluster-name \
    --log-type ACTION_LOGS
```

2. Configurez la destination de livraison en exécutant la commande suivante :

```
aws logs put-delivery-destination \
    --name my-ecs-logs-destination \
    --output-format json \
    --delivery-destination-configuration '{"destinationResourceArn":
 "arn:aws:logs:region:account-id:log-group:/aws/vendedlogs/ecs/action-logs/cluster-
name"}'
```

3. Créez la diffusion pour lier la source à la destination en exécutant la commande suivante :

```
aws logs create-delivery \
    --delivery-source-name my-ecs-action-logs \
    --delivery-destination-arn arn:aws:logs:region:account-id:delivery-
destination:my-ecs-logs-destination
```

Vérifier la transmission des journaux

Après avoir activé les journaux d'action, vérifiez que les journaux sont envoyés à votre destination.

Pour vérifier la livraison du journal

1. Déclenchez une action dans votre cluster. Par exemple, créez ou mettez à jour un service démon.
2. Ouvrez le CloudWatch console à https://console.aws.amazon.com/cloudwatch/.
3. Dans le panneau de navigation, choisissez Groupes de journaux.
4. Sélectionnez votre groupe de journaux Action Logs et vérifiez que de nouveaux flux de journaux apparaissent.

Supprimer les journaux d'actions

Vous pouvez supprimer les journaux d'actions à l'aide du AWS CLI ou console Amazon ECS.

Pour supprimer des journaux d'actions, votre identité IAM doit disposer des autorisations supplémentaires suivantes :

```
{
    "Effect": "Allow",
    "Action": [
        "logs:DeleteDelivery",
        "logs:DeleteDeliverySource",
        "logs:DeleteDeliveryDestination"
    ],
    "Resource": "*"
}
```

Pour supprimer les journaux d'actions à l'aide du AWS INTERFACE DE LIGNE DE COMMANDE (CLI)

1. Supprimez la livraison en exécutant la commande suivante :

   ```
   aws logs delete-delivery \
       --delivery-id delivery-id
   ```

2. Supprimez la source de diffusion en exécutant la commande suivante :

```
aws logs delete-delivery-source \
    --name my-ecs-action-logs
```

3. Supprimez la destination de livraison en exécutant la commande suivante :

```
aws logs delete-delivery-destination \
    --name my-ecs-logs-destination
```

Pour supprimer les journaux d'actions à l'aide de la console, accédez à la page des détails du cluster, choisissez Configuration onglet, puis choisissez Supprimer dans le Journaux d'actions section.

Étapes suivantes

Après avoir activé Action Logs, utilisez les ressources suivantes pour surveiller et dépanner vos clusters Amazon ECS :

- Résolution des problèmes liés aux journaux d'action Amazon ECS

Résolution des problèmes liés aux journaux d'action Amazon ECS

Les journaux d'action vous aident à diagnostiquer les problèmes en fournissant des enregistrements détaillés des opérations initiées par le service Amazon ECS. Chaque entrée de journal contient un horodatage, un niveau de journal, le nom de l'événement et une charge utile détaillée avec un contexte sur ce qui s'est passé et pourquoi. Cette rubrique couvre les scénarios de dépannage courants et fournit des informations prêtes à l'emploi CloudWatch Enregistre les requêtes Insights.

Comprendre les niveaux de journalisation

Les journaux d'action utilisent trois niveaux de journalisation pour indiquer la gravité de chaque événement :

`INFO`

Opérations normales telles que le lancement d’un déploiement, le lancement d’une tâche ou l’allocation d’une ressource. Ces événements indiquent qu’Amazon ECS exécute les actions prévues.

`WARN`

Non-fatal problèmes susceptibles de nécessiter une attention particulière, tels que les tentatives de nouvelle tentative ou les contraintes de capacité. Amazon ECS continue de progresser, mais l'opération pourrait prendre plus de temps que prévu.

`ERROR`

Défaillances nécessitant une action, telles qu'un échec de déploiement, une erreur d'allocation ou un démarrage de tâche défaillant. Passez en revue ces événements pour en identifier la cause racine et prendre des mesures correctives.

> **Tip**
>
> Commencez par filtrer pour `ERROR` ou `WARN` niveaux de journalisation pour identifier rapidement les problèmes. Vous pouvez ensuite étendre votre recherche à `INFO` événements pour un contexte supplémentaire.

Débogage d'un déploiement de service qui a échoué

Symptômes

Déploiement bloqué en cours ou automatiquement annulé.

Ce qu'il faut rechercher

Filtre pour `ERROR` journaux où `eventName` contient `DEPLOYMENT`. Vérifiez le `detail` charge utile pour l'ARN de déploiement, raison de l'état et nombre de tâches ayant échoué.

L'exemple suivant montre un échec de déploiement où aucun candidat au rollback n'était disponible :

```
{
  "resourceArn": "arn:aws:ecs:us-east-1:111122223333:cluster/my-cluster",
  "actionSourceId": "service/my-cluster/my-service",
  "logLevel": "ERROR",
  "eventTimestamp": 1784573730986,
  "detail": {
    "statusReason": "No rollback candidate was found to run the rollback.",
    "serviceDeploymentArn": "arn:aws:ecs:us-east-1:111122223333:service-deployment/my-
cluster/my-service/abc123def456",
    "status": "FAILED",
```

```
    "eventName": "SERVICE_DEPLOYMENT_ROLLBACK_FAILED"
  }
}
```

Résolution

Vérifiez le `detail.statusReason` champ pour identifier les raisons pour lesquelles les tâches ont échoué. Utilisez ce qui suit CloudWatch Requête Logs Insights pour trouver toutes les erreurs de déploiement :

```
fields @timestamp, detail.statusReason
| filter logLevel = "ERROR" and detail.deploymentArn like /my-cluster\/my-service/
| sort @timestamp desc
| limit 20
```

Diagnostic des problèmes liés aux démons gérés

Symptômes

Le démon ne démarre pas sur les instances ou le déploiement du démon est bloqué.

Ce qu'il faut rechercher

Filtrez les journaux en fonction de l'ARN du démon pour trouver les événements liés à votre démon. Vous pouvez également filtrer par `logLevel` pour limiter les résultats aux avertissements ou aux erreurs.

L'exemple suivant montre une tâche de démon qui n'a pas pu démarrer en raison d'une mémoire insuffisante sur l'instance cible :

```
{
  "timestamp": 1719500050000,
  "logLevel": "WARN",
  "account": "123456789012",
  "region": "us-east-1",
  "resourceArn": "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster",
  "actionSourceId": "arn:aws:ecs:us-east-1:123456789012:daemon/my-cluster/my-logging-
daemon",
  "eventName": "DAEMON_TASK_START_IMPAIRED",
  "detail": {
    "statusReason": "RESOURCE:MEMORY - Unable to place daemon task on container
 instance: insufficient memory",
```

```
    "containerInstanceArn": "arn:aws:ecs:us-east-1:123456789012:container-instance/my-
cluster/a1b2c3d4"
  }
}
```

Résolution

Vérifiez que les instances cibles disposent de suffisamment de ressources pour exécuter la tâche du démon. Si les instances sont épuisées, le démon risque de ne pas démarrer tant que de nouvelles instances ne seront pas disponibles. Utilisez la requête suivante pour filtrer les événements d'un démon spécifique :

```
fields @timestamp, logLevel, detail.statusReason
| filter actionSourceId like /my-logging-daemon/
| filter logLevel in ["ERROR", "WARN"]
| sort @timestamp desc
| limit 20
```

Comprendre les raisons de l'arrêt des tâches du démon

Symptômes

Les tâches du démon s'arrêtent de façon inattendue.

Ce qu'il faut rechercher

Filtre pour `WARN` et `ERROR` journaux contenant des informations détaillées sur les échecs des tâches du démon. Les champs suivants du `detail` la charge utile fournit des informations sur les raisons pour lesquelles une tâche de démon s'est arrêtée :

`detail.stopCode`

Code lisible par machine qui classe la raison de l'arrêt de la tâche.

`detail.statusReason`

Description lisible par l'homme de la raison pour laquelle la tâche s'est arrêtée. Ce champ n'est pas présent dans toutes les entrées du journal.

`detail.containerStoppedReason`

Informations contextuelles supplémentaires sur le conteneur spécifique à l'origine de l'arrêt de la tâche.

Pour obtenir la liste complète des codes d'arrêt des tâches et leurs descriptions, voir Messages d'erreurs liées aux tâches Amazon ECS arrêtées.

> **Note**
>
> La tâche du démon attendue s'arrête, telle que `UserInitiated`, sont filtrés à partir des journaux d'actions. Seuls les arrêts de tâches inattendus ou liés à des erreurs apparaissent.

Utile CloudWatch Requêtes Logs Insights

Utilisez ce qui suit CloudWatch Enregistre les requêtes Insights afin d'étudier les problèmes courants liés à vos services Amazon ECS.

Toutes les erreurs survenues au cours de la dernière heure

```
fields @timestamp, logLevel, eventName, detail.statusReason
| filter logLevel = "ERROR"
| sort @timestamp desc
| limit 50
```

> **Note**
>
> Le `detail.statusReason` le champ n'est pas présent dans toutes les entrées du journal. Si le champ est vide dans vos résultats, utilisez `eventName` et dans d'autres domaines du `detail` charge utile pour le contexte.

Événements pour un service spécifique

```
fields @timestamp, logLevel, detail.statusReason
| filter @message like /my-service/
| sort @timestamp desc
| limit 50
```

Calendrier de déploiement pour un déploiement spécifique

```
fields @timestamp, logLevel, detail.statusReason
| filter detail.deploymentArn like /my-cluster\/my-service\/abc123/
| sort @timestamp asc
```

Les échecs des tâches du démon sont regroupés par raison

```
filter logLevel = "ERROR" and detail.stopCode != ""
| stats count(*) as failureCount by detail.stopCode, detail.statusReason
| sort failureCount desc
| limit 20
```

Surveillance des applications à l'aide des métadonnées Amazon ECS

Vous pouvez utiliser les métadonnées des tâches et des conteneurs pour résoudre les problèmes liés à vos charges de travail et apporter des modifications de configuration en fonction de l'environnement d'exécution.

Les métadonnées incluent les catégories suivantes :

- Task-level attributs qui fournissent des informations sur l'endroit où la tâche est en cours d'exécution.
- Container-level attributs qui fournissent l'ID Docker, le nom et les détails de l'image.

  Cela permet de voir l'intérieur du conteneur.
- Paramètres réseau tels que les adresses IP, les sous-réseaux et le mode réseau.

  Cela facilite la configuration et le dépannage du réseau.
- Statut et état de la tâche

  Cela vous permet de savoir si les tâches sont en cours d'exécution.

Pour afficher les métadonnées, utilisez l'une des méthodes suivantes :

- Fichier de métadonnées de conteneur

  À compter de la version 1.15.0 de l'agent conteneur Amazon ECS, diverses métadonnées de conteneur sont disponibles dans les conteneurs ou dans l'instance de conteneur hôte. En activant cette fonctionnalité, vous pouvez demander des informations sur une tâche, un conteneur ou une instance de conteneur depuis le conteneur ou l'instance de conteneur hôte. Le fichier de métadonnées est créé sur l'instance hôte et monté dans le conteneur en tant que volume Docker et n'est donc pas disponible lorsqu'une tâche est hébergée sur AWS Fargate.

- Point de terminaison des métadonnées de tâches

  L'agent de conteneur Amazon ECS injecte une variable d'environnement dans chaque conteneur, appelée point de terminaison des métadonnées de tâches, qui fournit diverses métadonnées de tâches et statistiques Docker au conteneur.

- Introspection de conteneur

  L'agent de conteneur Amazon ECS fournit une opération API pour l'obtention d'informations sur l'instance de conteneur sur laquelle l'agent est exécuté et sur les tâches connexes elles aussi exécutées sur cette instance.

Variables d’environnement Amazon ECS

Les variables d’environnement contrôlent le comportement d’une tâche, notamment la capacité utilisée pour exécuter la tâche. Amazon ECS définit les variables d’environnement suivantes pour vos tâches :

- `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` : le point de terminaison URL HTTP complet que le kit SDK doit utiliser lors d’une requête d’informations d’identification. Cela inclut à la fois le schéma et l’hôte. Pour plus d’informations, consultez la section Fournisseur d’informations d’identification du conteneur dans le Guide de référence des kits SDK AWS .
- `ECS_CONTAINER_METADATA_URI_V4` : l’adresse de la version 4 des métadonnées de la tâche. Pour plus d’informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4.
- `ECS_CONTAINER_METADATA_URI` : l’adresse de la version 3 des métadonnées de la tâche. Pour plus d’informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 3.
- `ECS_AGENT_URI` : l’adresse de base pour les différents points de terminaison pris en charge par Fargate. Pour plus d’informations, consultez :
  - Point de terminaison de protection évolutive des tâches Amazon ECS
  - Points de terminaison d’injection de pannes Amazon ECS
- `AWS_EXECUTION_ENV` : les informations relatives à l’option de calcul sur laquelle la tâche s’exécute.
  - Pour Fargate, Amazon ECS définit ce paramètre sur `AWS_ECS_FARGATE`.
  - Pour EC2, Amazon ECS définit ce paramètre sur `AWS_ECS_EC2`.

- `AWS_DEFAULT_REGION` — La valeur par défaut Région AWS pour votre Compte AWS. Il s'agit de la région par défaut dans laquelle votre tâche s'exécute.
- `AWS_REGION` : la région dans laquelle la tâche s'exécute. Si elle est définie, cette valeur remplace `AWS_DEFAULT_REGION`.

Vous pouvez consulter les variables d'environnement dans les métadonnées de tâche. Pour plus d'informations, consultez l'une des rubriques suivantes :

- Fargate : Métadonnées de tâches Amazon ECS disponibles pour les tâches sur Fargate
- EC2 : Métadonnées de tâches disponibles pour les tâches sur EC2

Fichier de métadonnées de conteneur Amazon ECS

À compter de la version 1.15.0 de l'agent conteneur Amazon ECS, diverses métadonnées de conteneur sont disponibles dans les conteneurs ou dans l'instance de conteneur hôte. En activant cette fonctionnalité, vous pouvez demander des informations sur une tâche, un conteneur ou une instance de conteneur depuis le conteneur ou l'instance de conteneur hôte. Le fichier de métadonnées est créé sur l'instance hôte et monté dans le conteneur en tant que volume Docker. Il n'est donc pas disponible lorsqu'une tâche est hébergée sur AWS Fargate.

Le fichier de métadonnées de conteneur est nettoyé sur l'instance de l'hôte lors du nettoyage du conteneur. Vous pouvez définir quand cette opération doit se produire avec la variable `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` de l'agent de conteneur. Pour plus d'informations, consultez Nettoyage automatique des tâches et des images Amazon ECS.

Rubriques

- Emplacements des fichiers de métadonnées de conteneur
- Activation des métadonnées de conteneur Amazon ECS
- Format du fichier de métadonnées de conteneur Amazon ECS

Emplacements des fichiers de métadonnées de conteneur

Par défaut, le fichier de métadonnées de conteneur est écrit sur les chemins d'hôte et de conteneur suivants.

- Pour les instances Linux :

- Chemin hôte :`/var/lib/ecs/data/metadata/`*`cluster_name`*`/`*`task_id`*`/`*`container_name`*`/ecs-container-metadata.json`

  > **Note**
  >
  > Le chemin hôte Linux suppose que le chemin de montage du répertoire de données par défaut (`/var/lib/ecs/data`) est utilisé lorsque l'agent est démarré. Si vous n'utilisez pas d'Amazon ECS-optimized AMI (ou `ecs-init` package pour démarrer et maintenir le conteneur (agent), assurez-vous de régler le `ECS_HOST_DATA_DIR` variable de configuration de l'agent sur le chemin de l'hôte où se trouve le fichier d'état de l'agent de conteneur. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

- Chemin du conteneur : `/opt/ecs/metadata/`*`random_ID`*`/ecs-container-metadata.json`

- Pour les instances Windows :
  - Chemin hôte :`C:\ProgramData\Amazon\ECS\data\metadata\`*`task_id`*`\`*`container_name`*`\ecs-container-metadata.json`
  - Chemin du conteneur : `C:\ProgramData\Amazon\ECS\metadata\`*`random_ID`*`\ecs-container-metadata.json`

Toutefois, pour en faciliter l'accès, l'emplacement du fichier de métadonnées de conteneur correspond à la variable d'environnement `ECS_CONTAINER_METADATA_FILE` à l'intérieur du conteneur. Vous pouvez lire le contenu du fichier depuis le conteneur via la commande suivante :

- Pour les instances Linux :

```
cat $ECS_CONTAINER_METADATA_FILE
```

- Pour les instances Windows (PowerShell) :

```
Get-Content -path $env:ECS_CONTAINER_METADATA_FILE
```

Activation des métadonnées de conteneur Amazon ECS

Vous pouvez activer les métadonnées de conteneur au niveau de l'instance de conteneur en définissant la variable `ECS_ENABLE_CONTAINER_METADATA` de l'agent de conteneur sur `true`. Vous pouvez spécifier cette variable dans le fichier de configuration `/etc/ecs/ecs.config` et redémarrer l'agent. Vous pouvez également la configurer sous forme de variable d'environnement Docker au moment de l'exécution lorsque l'agent de conteneur est démarré. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Si la valeur `ECS_ENABLE_CONTAINER_METADATA` est définie sur `true` au démarrage de l'agent, des fichiers de métadonnées sont créés pour tous les conteneurs générés à partir de cet instant. L'agent de conteneur Amazon ECS ne peut pas créer de fichiers de métadonnées pour les conteneurs qui ont été créés avant que la variable d'agent de conteneur `ECS_ENABLE_CONTAINER_METADATA` n'ait été définie sur `true`. Pour vous assurer que tous les conteneurs reçoivent des fichiers de métadonnées, vous devez définir cette variable d'agent au lancement de l'instance de conteneur. Voici un exemple de script de données utilisateur qui définit cette variable et enregistre votre instance de conteneur auprès de votre cluster.

```
#!/bin/bash
cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=your_cluster_name
ECS_ENABLE_CONTAINER_METADATA=true
EOF
```

Format du fichier de métadonnées de conteneur Amazon ECS

Les informations suivantes sont stockées dans le fichier JSON de métadonnées de conteneur.

`Cluster`

Nom du cluster sur lequel le conteneur de la tâche est en cours d'exécution.

`ContainerInstanceARN`

Amazon Resource Name (ARN) de l'instance de conteneur hôte.

`TaskARN`

Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`TaskDefinitionFamily`

Nom de la famille de définitions de tâche utilisée par le conteneur.

`TaskDefinitionRevision`

Révision de définition de tâche utilisée par le conteneur.

`ContainerID`

ID de conteneur Docker (et non l'ID de conteneur Amazon ECS).

`ContainerName`

Nom de conteneur issu de la définition de tâche Amazon ECS.

`DockerContainerName`

Nom de conteneur que le démon Docker utilise pour le conteneur (par exemple, nom qui s'affiche dans la sortie de la commande docker ps).

`ImageID`

Résumé SHA de l'image Docker utilisée pour lancer le conteneur.

`ImageName`

Nom et balise de l'image Docker utilisée pour lancer le conteneur.

`PortMappings`

Mappages de port associés au conteneur.

`ContainerPort`

Port du conteneur qui est exposé.

`HostPort`

Port de l'instance de conteneur hôte qui est exposé.

`BindIp`

Adresse IP de liaison affectée au conteneur par Docker. Cette adresse IP est uniquement appliquée avec le mode réseau `bridge`. Elle est uniquement accessible à partir de l'instance de conteneur.

`Protocol`

Protocole réseau utilisé pour le mappage de port.

`Networks`

Mode réseau et adresse IP du conteneur.

`NetworkMode`

Mode réseau de la tâche à laquelle le conteneur appartient.

`IPv4Addresses`

Adresses IP associées au conteneur.

> **Important**
>
> Si votre tâche utilise le mode réseau `awsvpc`, l'adresse IP du conteneur n'est pas renvoyée. Dans ce cas, vous pouvez récupérer l'adresse IP en lisant le/etc/hosts fichier à l'aide de la commande suivante :
>
> ```
> tail -1 /etc/hosts | awk '{print $1}'
> ```

`MetadataFileStatus`

Statut du fichier de métadonnées. Lorsque le statut est `READY`, le fichier de métadonnées est actuel et complet. Si le fichier n'est pas encore prêt (par exemple, au moment où la tâche est démarrée), une version tronquée du format de fichier est disponible. Pour éviter toute condition de concurrence où le conteneur démarrerait alors que les métadonnées ne sont pas encore écrites, vous pouvez analyser le fichier de métadonnées et attendre que ce paramètre passe à `READY` au préalable, selon les métadonnées. Cela est généralement disponible en moins de 1 seconde à partir du moment où le conteneur démarre.

`AvailabilityZone`

Zone de disponibilité dans laquelle réside l'instance de conteneur hôte.

`HostPrivateIPv4Address`

Adresse IP privée de la tâche à laquelle appartient le conteneur.

`HostPublicIPv4Address`

Adresse IP publique de la tâche à laquelle appartient le conteneur.

Example Fichier de métadonnées du conteneur Amazon ECS (**PRÊT**)

L'exemple suivant montre un conteneur de fichiers de métadonnées avec le statut READY.


```
{
    "Cluster":"arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
    "TaskARN":"arn:aws:ecs:us-east-1:123456789012:task/MyCluster/
b593651c4d6b44a6b2b583f45c957e15",
    "Family":"curltest-container",
    "Revision":"2",
    "DesiredStatus":"RUNNING",
    "KnownStatus":"RUNNING",
    "Limits":
        {
            "CPU":0.25,
            "Memory":512
        },
    "PullStartedAt":"2025-01-17T20:56:17.394610044Z",
    "PullStoppedAt":"2025-01-17T20:56:25.282708213Z",
    "AvailabilityZone":"us-east-1b",
    "LaunchType":"FARGATE",
    "Containers":[
        {
            "DockerId":"b593651c4d6b44a6b2b583f45c957e15-3356213583",
            "Name":"curltest","DockerName":"curltest",
            "Image":"public.ecr.aws/amazonlinux/amazonlinux:latest",

 "ImageID":"sha256:7f371357694782356b65c7fd60dd1ca124c47bd5ed1b1ffe7c0e17f562898367",
            "Labels":
                {
                    "com.amazonaws.ecs.cluster":"arn:aws:ecs:us-
east-1:123456789012:cluster/MyCluster",
                    "com.amazonaws.ecs.container-name":"curltest",
                    "com.amazonaws.ecs.task-arn":"arn:aws:ecs:us-
east-1:123456789012:task/MyCluster/b593651c4d6b44a6b2b583f45c957e15",
                    "com.amazonaws.ecs.task-definition-family":"curltest-
container","com.amazonaws.ecs.task-definition-version":"2"
                },
            "DesiredStatus":"RUNNING",
            "KnownStatus":"RUNNING",
            "Limits":
                {
                    "CPU":2
                },
```

```
            "CreatedAt":"2025-01-17T20:56:26.180347056Z",
            "StartedAt":"2025-01-17T20:56:26.180347056Z",
            "Type":"NORMAL",
            "LogDriver":"awslogs",
            "LogOptions":
                {
                    "awslogs-create-group":"true",
                    "awslogs-group":"/ecs/curltest-container",
                    "awslogs-region":"us-east-1",
                    "awslogs-stream":"ecs/curltest/b593651c4d6b44a6b2b583f45c957e15"
                       },
            "ContainerARN":"arn:aws:ecs:us-east-1:123456789012:container/MyCluster/
b593651c4d6b44a6b2b583f45c957e15/934575e8-5bdb-478f-b763-2341a85b690e",
            "Networks":[
                {
                    "NetworkMode":"awsvpc",
                    "IPv4Addresses":["10.0.1.58"]
                }
            ],
            "Snapshotter":"overlayfs"
        }
    ],
    "ClockDrift":
        {
            "ClockErrorBound":0.487801,"ReferenceTimestamp":"2025-01-17T20:56:02Z",
            "ClockSynchronizationStatus":"SYNCHRONIZED"
        },
    "FaultInjectionEnabled":false
}
```

Example Fichier de métadonnées du conteneur Amazon ECS incomplet (pas encore) **PRÊT**)

L'exemple suivant montre un fichier de métadonnées de conteneur qui n'a pas encore atteint le statut READY. Les informations contenues dans le fichier sont limitées à quelques paramètres connus à partir de la définition de tâche. Le fichier de métadonnées de conteneur doit être prêt 1 seconde à compter du démarrage du conteneur.

```
{
    "Cluster": "default",
    "ContainerInstanceARN": "arn:aws:ecs:us-west-2:012345678910:container-instance/
default/1f73d099-b914-411c-a9ff-81633b7741dd",
    "TaskARN": "arn:aws:ecs:us-west-2:012345678910:task/default/
d90675f8-1a98-444b-805b-3d9cabb6fcd4",
```

```
    "ContainerName": "metadata"
}
```

Métadonnées de tâche Amazon ECS disponibles pour les tâches sur les instances gérées Amazon ECS

Amazon ECS sur instances gérées Amazon ECS fournit une méthode pour récupérer diverses métadonnées, métriques réseau et statistiques Docker relatives à vos conteneurs et aux tâches dont ils font partie. On parle alors du point de terminaison des métadonnées de tâches. La version 4 du point de terminaison des métadonnées de tâche est disponible pour les tâches Amazon ECS sur instances gérées Amazon ECS.

Tous les conteneurs appartenant à des tâches qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Rubriques

- Point de terminaison des métadonnées de tâche Amazon ECS version 4 pour les tâches sur instances gérées Amazon ECS

Point de terminaison des métadonnées de tâche Amazon ECS version 4 pour les tâches sur instances gérées Amazon ECS

> **Important**
>
> Si vous utilisez des tâches Amazon ECS hébergées sur des instances Amazon EC2, consultez la section Point de terminaison des métadonnées de tâche Amazon ECS.

À partir des instances gérées Amazon ECS, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI_V4` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 4, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches.

Le point de terminaison des métadonnées de tâche est activé par défaut pour toutes les tâches Amazon ECS exécutées sur des instances gérées Amazon ECS.

> **Note**
>
> Pour éviter de créer d'autres versions de point de terminaison de métadonnées de tâche à l'avenir, des métadonnées supplémentaires peuvent être ajoutées à la sortie de la version 4. Nous ne supprimons pas les métadonnées existantes et ne modifions pas les noms des champs de métadonnées.

Chemins d’accès au point de terminaison des métadonnées de tâche des instances gérées Amazon ECS version 4

Les points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI_V4}`

Ce chemin renvoie des métadonnées pour le conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Ce chemin renvoie les métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâche Amazon ECS v4 pour les tâches sur les instances gérées Amazon ECS.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Ce chemin renvoie les statistiques Docker pour le conteneur Docker. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

> **Note**
>
> Les tâches Amazon ECS sur instances gérées Amazon ECS exigent que le conteneur s’exécute pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

> **Note**
>
> Les tâches Amazon ECS sur instances gérées Amazon ECS exigent que le conteneur s'exécute pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/tasks`

Ce chemin renvoie les métadonnées de toutes les tâches exécutées sur l'instance de conteneur. La réponse est un tableau JSON d'objets de métadonnées de tâches, où chaque élément a la même structure que la réponse du `/task` point final. Pour plus d'informations sur la réponse à chaque tâche, voir Réponse JSON des métadonnées de tâche Amazon ECS v4 pour les tâches sur les instances gérées Amazon ECS.

> **Note**
>
> Ce point de terminaison n'est disponible que pour les tâches exécutées en tant que service de démon géré sur les instances gérées Amazon ECS.

`${ECS_CONTAINER_METADATA_URI_V4}/tasks/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs pour toutes les tâches exécutées sur l'instance de conteneur. La réponse est un tableau JSON où chaque élément contient une carte des ID de conteneurs avec leurs statistiques Docker pour une tâche donnée. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

> **Note**
>
> Ce point de terminaison n'est disponible que pour les tâches exécutées en tant que service de démon géré sur les instances gérées Amazon ECS.

Réponse JSON des métadonnées de tâche Amazon ECS v4 pour les tâches sur les instances gérées Amazon ECS

Les métadonnées suivantes sont renvoyées dans la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI_V4}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`ServiceName`

Nom du service auquel appartient la tâche. ServiceName apparaîtra si la tâche est associée à un service.

`VPCID`

L'Id de VPC de l'instance de conteneur.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

`LaunchType`

Type de lancement utilisé par la tâche. Pour les tâches exécutées sur les instances gérées Amazon ECS, le type de lancement est `MANAGED_INSTANCES`.

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

Lorsque vous utilisez Fargate, l’ID est un hexadécimal de 32 chiffres suivi d’un nombre à 10 chiffres.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`LogDriver`

Pilote de journal utilisé par le conteneur.

`LogOptions`

Options de pilote de journal définies pour le conteneur.

`ContainerARN`

L'Amazon Resource Name (ARN) du conteneur.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`Snapshotter`

Le snapshotter qui a été utilisé par containerd pour télécharger cette image de conteneur. Les valeurs valides sont `overlayfs`, qui est la valeur par défaut, et `soci`, utilisées en cas de chargement différé avec un index SOCI.

`RestartCount`

Nombre de fois où le conteneur a été redémarré.

> **Note**
>
> Les métadonnées `RestartCount` ne sont incluses que si une politique de redémarrage est activée pour le conteneur. Pour plus d'informations, consultez Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cette fonctionnalité utilise le Service de synchronisation temporelle d'Amazon pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations, consultez la section Réglage de l'heure pour votre instance Linux dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

`FaultInjectionEnabled`

Indique si l'injection de pannes est activée pour la tâche. Pour plus d'informations, consultez Utilisation de l'injection de pannes avec vos charges de travail Amazon ECS et Fargate.

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur instances gérées Amazon ECS

Les exemples suivants présentent des sorties de points de terminaison des métadonnées de tâche pour les tâches Amazon ECS exécutées sur des instances gérées Amazon ECS.

Depuis le conteneur, vous pouvez utiliser curl suivi du point de terminaison des métadonnées de la tâche pour interroger le point de terminaison, par exemple `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`.

Exemple de réponse de métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de sortie.

```
{
  "DockerId": "400c0466d1e54c0691aae0f86e0f9dc6-2531612879",
  "Name": "nginx",
  "DockerName": "nginx",
  "Image": "public.ecr.aws/nginx/nginx:latest"
  "ImageID": "sha256:d5f28ef21aabddd098f3dbc21fe5b7a7d7a184720bc07da0b6c9b9820e97f25e",
  "Labels": {
    "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/managed-
instances-cluster",
    "com.amazonaws.ecs.container-name": "curl",
```

```
    "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/managed-
instances-cluster/400c0466d1e54c0691aae0f86e0f9dc6",
    "com.amazonaws.ecs.task-definition-family": "ecs-managed-instances-task-def",
    "com.amazonaws.ecs.task-definition-version": "7"
  },
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": { "CPU": 2 },
  "CreatedAt": "2025-09-24T19:39:04.88336233Z",
  "StartedAt": "2025-09-24T19:39:04.88336233Z",
  "Type": "NORMAL",
  "LogDriver": "awslogs",
  "LogOptions": {
    "awslogs-create-group": "true",
    "awslogs-group": "/ecs/managed-instances-task-def",
    "awslogs-region": "us-west-2",
    "awslogs-stream": "ecs/nginx/400c0466d1e54c0691aae0f86e0f9dc6"
  },
  "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/managed-instances-
cluster/400c0466d1e54c0691aae0f86e0f9dc6/3703f0e4-a351-4f1e-a0e7-6981ea7adc8b",
  "Networks": [
    {
      "NetworkMode": "awsvpc",
      "IPv4Addresses": ["172.31.62.223"],
      "AttachmentIndex": 0,
      "MACAddress": "0e:41:0b:1e:f2:fb",
      "IPv4SubnetCIDRBlock": "172.31.48.0/20",
      "PrivateDNSName": "ip-172-31-62-223.us-west-2.compute.internal",
      "SubnetGatewayIpv4Address": "172.31.48.1/20"
    }
  ],
  "Snapshotter": "overlayfs"
}
```

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur instances gérées Amazon ECS

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
  "Cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/managed-instances-cluster",
```

```
  "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/managed-instances-
cluster/400c0466d1e54c0691aae0f86e0f9dc6",
  "Family": "managed-instances-task-def",
  "Revision": "7",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": { "CPU": 1, "Memory": 3072 },
  "PullStartedAt": "2025-09-24T19:38:58.682942001Z",
  "PullStoppedAt": "2025-09-24T19:39:03.091597524Z",
  "AvailabilityZone": "us-west-2d",
  "LaunchType": "MANAGED_INSTANCES",
  "Containers": [
    {
      "DockerId": "400c0466d1e54c0691aae0f86e0f9dc6-2531612879",
      "Name": "curl",
      "DockerName": "curl",
      "Image": "nginx",
      "ImageID":
 "sha256:d5f28ef21aabddd098f3dbc21fe5b7a7d7a184720bc07da0b6c9b9820e97f25e",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
managed-instances-cluster",
        "com.amazonaws.ecs.container-name": "nginx",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/managed-
instances-cluster/400c0466d1e54c0691aae0f86e0f9dc6",
        "com.amazonaws.ecs.task-definition-family": "managed-instances-task-def",
        "com.amazonaws.ecs.task-definition-version": "7"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2025-09-24T19:39:04.88336233Z",
      "StartedAt": "2025-09-24T19:39:04.88336233Z",
      "Type": "NORMAL",
      "LogDriver": "awslogs",
      "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/managed-instances-task-def",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/nginx/400c0466d1e54c0691aae0f86e0f9dc6"
      },
      "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/managed-instances-
cluster/400c0466d1e54c0691aae0f86e0f9dc6/3703f0e4-a351-4f1e-a0e7-6981ea7adc8b",
      "Networks": [
```

```
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.62.223"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:41:0b:1e:f2:fb",
          "IPv4SubnetCIDRBlock": "172.31.48.0/20",
          "PrivateDNSName": "ip-172-31-62-223.us-west-2.compute.internal",
          "SubnetGatewayIpv4Address": "172.31.48.1/20"
        }
      ],
      "Snapshotter": "overlayfs"
    }
  ],
  "ServiceName": "exec-service-2",
  "ClockDrift": {
    "ClockErrorBound": 0.14120749999999999,
    "ReferenceTimestamp": "2025-09-24T19:48:37Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "FaultInjectionEnabled": false
}
```

Exemple de réponse des statistiques de tâche

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, les métriques réseau dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
  "400c0466d1e54c0691aae0f86e0f9dc6-2531612879": {
    "read": "2025-09-24T19:51:54.899736614Z",
    "preread": "2025-09-24T19:51:44.901199024Z",
    "pids_stats": {},
    "blkio_stats": {
      "io_service_bytes_recursive": [
        { "major": 259, "minor": 1, "op": "read", "value": 0 },
        { "major": 259, "minor": 1, "op": "write", "value": 131072 },
        { "major": 259, "minor": 0, "op": "read", "value": 48173056 },
        { "major": 259, "minor": 0, "op": "write", "value": 0 },
        { "major": 252, "minor": 0, "op": "read", "value": 48173056 },
        { "major": 252, "minor": 0, "op": "write", "value": 0 }
      ],
      "io_serviced_recursive": null,
```

```
      "io_queue_recursive": null,
      "io_service_time_recursive": null,
      "io_wait_time_recursive": null,
      "io_merged_recursive": null,
      "io_time_recursive": null,
      "sectors_recursive": null
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
        "total_usage": 670462000,
        "usage_in_kernelmode": 276072000,
        "usage_in_usermode": 394389000
      },
      "system_cpu_usage": 787660000000,
      "online_cpus": 1,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 663809000,
        "usage_in_kernelmode": 273333000,
        "usage_in_usermode": 390476000
      },
      "system_cpu_usage": 777710000000,
      "online_cpus": 1,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "memory_stats": {
      "usage": 83562496,
      "stats": {
        "active_anon": 24576,
        "active_file": 258048,
        "anon": 32264192,
        "anon_thp": 0,
```

```
        "file": 48533504,
        "file_dirty": 0,
        "file_mapped": 36286464,
        "file_writeback": 0,
        "inactive_anon": 32243712,
        "inactive_file": 48271360,
        "kernel_stack": 442368,
        "pgactivate": 6,
        "pgdeactivate": 0,
        "pgfault": 18936,
        "pglazyfree": 0,
        "pglazyfreed": 0,
        "pgmajfault": 311,
        "pgrefill": 0,
        "pgscan": 0,
        "pgsteal": 0,
        "shmem": 4096,
        "slab": 1598960,
        "slab_reclaimable": 1080040,
        "slab_unreclaimable": 518920,
        "sock": 0,
        "thp_collapse_alloc": 0,
        "thp_fault_alloc": 0,
        "unevictable": 0,
        "workingset_activate": 0,
        "workingset_nodereclaim": 0,
        "workingset_refault": 0
      },
      "limit": 18446744073709551615
    },
    "name": "nginx",
    "id": "400c0466d1e54c0691aae0f86e0f9dc6-2531612879"
  }
}
```


Métadonnées de tâches disponibles pour les tâches sur EC2

L'agent de conteneur Amazon ECS fournit une méthode pour récupérer les diverses métadonnées de tâches et les statistiques Docker. On parle alors du point de terminaison des métadonnées de tâches. Les versions suivantes sont disponibles :

- Point de terminaison des métadonnées de tâche version 4 : fournit une variété de métadonnées et de statistiques Docker aux conteneurs. Peut également fournir des données de débit réseau.

Disponible pour les tâches Amazon ECS lancées sur des instances Linux Amazon EC2 exécutant au moins la version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4.

- Point de terminaison des métadonnées de tâche version 3 : fournit une variété de métadonnées et de statistiques Docker aux conteneurs. Disponible pour les tâches Amazon ECS lancées sur des instances Linux Amazon EC2 exécutant au moins la version `1.21.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 3.
- Point de terminaison de métadonnées de tâche version 2 : disponible pour les tâches Amazon ECS lancées sur des instances Linux Amazon EC2 exécutant au moins la version `1.17.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 2.

Si votre tâche Amazon ECS est hébergée sur Amazon EC2, ou si votre tâche utilise le `host` en mode réseau et est hébergé sur des instances gérées Amazon ECS, vous pouvez également accéder aux métadonnées de l'hôte de tâches à l'aide du Point de terminaison du service de métadonnées d'instance (IMDS). La commande suivante, lorsqu'elle est exécutée depuis l'instance hébergeant la tâche, répertorie l'ID de l'instance hôte.

```
curl http://169.254.169.254/latest/meta-data/instance-id
```

Si votre tâche Amazon ECS est hébergée sur Amazon EC2 et dans un IPv6-only configuration, vous pouvez accéder aux métadonnées de l'hôte de tâches à l'aide du point de terminaison IPv6 IMDS. La commande suivante, lorsqu'elle est exécutée depuis l'instance hébergeant la tâche, répertorie l'ID de l'instance hôte sur IPv6.

```
curl http://[fd00:ec2::254]/latest/meta-data/instance-id
```

Pour accéder au point de terminaison IMDS IPv6, activez le point de terminaison IMDS IPv6 sur votre instance de conteneur et configurez également le mode de point de terminaison du service de métadonnées à l'aide du fournisseur d'informations d'identification IMDS pour le kit SDK de votre

choix vers `IPv6`. Pour plus d'informations sur l'activation du point de terminaison IMDS IPv6 pour votre instance de conteneur, consultez la section Configurer les options du service de métadonnées d'instance dans le Guide de l'utilisateur Amazon EC2. Pour plus d'informations sur le fournisseur d'informations d'identification IMDS pour les SDK, consultez la section Fournisseur d'informations d'identification IMDS dans le Guide de référence des kits SDK et des outils AWS .

> **Note**
>
> Le point de terminaison IMDS IPv6 n'est pas accessible lorsque le paramètre du compte `awsvpcTrunking` est activé. Pour accéder aux informations d'identification du rôle IAM de l'instance de conteneur lorsque `awsvpcTrunking` est activée, vous pouvez utiliser un rôle IAM de tâche à la place. Pour plus d'informations sur les rôles IAM, consultez la section rôle IAM de tâche Amazon ECS.

Les informations que vous pouvez obtenir à partir du point de terminaison sont divisées en catégories telles que *`instance-id`*. Pour plus d'informations sur les différentes catégories de métadonnées d'instance hôte que vous pouvez obtenir à l'aide du point de terminaison, consultez la section Catégories de métadonnées d'instance.

Point de terminaison des métadonnées de tâches Amazon ECS version 4

L'agent de conteneur Amazon ECS injecte une variable d'environnement dans chaque conteneur, appelée point de terminaison des métadonnées de tâches, qui fournit diverses métadonnées de tâches et statistiques Docker au conteneur.

Les métadonnées des tâches et les statistiques de débit du réseau sont envoyées à CloudWatch Container Insights et peut être consulté dans Console de gestion AWS. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.

> **Note**
>
> Amazon ECS fournit des versions antérieures du point de terminaison des métadonnées de tâches. Pour éviter de créer d'autres versions de point de terminaison de métadonnées de tâche à l'avenir, des métadonnées supplémentaires peuvent être ajoutées à la sortie de la version 4. Nous ne supprimons pas les métadonnées existantes et ne modifions pas les noms des champs de métadonnées.

La variable d'environnement est injectée par défaut dans les conteneurs des tâches Amazon ECS lancées sur les instances Linux Amazon EC2 exécutant au moins la version `1.39.0` de l'agent de conteneur Amazon ECS. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Pour plus d'informations, consultez Gestion des instances de conteneur Amazon ECS Linux.

> **Note**
>
> Vous pouvez ajouter la prise en charge de cette fonctionnalité sur les instances Amazon EC2 utilisant des versions plus anciennes de l'agent de conteneur Amazon ECS en mettant à jour l'agent à la dernière version. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Pour des exemples de sortie de métadonnées de tâches, consultez la section Exemples de métadonnées de tâches Amazon ECS v4.

Chemins des points de terminaison des métadonnées de tâches version 4

Les chemins des points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI_V4}`

Ce chemin renvoie des métadonnées pour le conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Ce chemin renvoie les métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâches Amazon ECS v4.

`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API `ListTagsForResource`. Toutes les erreurs reçues lors de la récupération des métadonnées de balise seront incluses dans le champ `Errors` de la réponse.

> **Note**
>
> Le champ `Errors` est uniquement dans la réponse pour les tâches hébergées sur des instances Linux Amazon EC2 exécutant au moins la version `1.50.0` de l'agent de conteneur. Pour les instances Windows Amazon EC2 qui utilisent le mode réseau `awsvpc`, la version de l'agent de conteneur Amazon ECS doit être au moins `1.54.0`. Ce point de terminaison nécessite l'autorisation `ecs.ListTagsForResource`.

> **Important**
>
> Lorsque vous utilisez le point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, sachez que chaque appel génère jusqu'à deux requêtes d'API `ecs:ListTagsForResource` (une pour les balises d'instance de conteneur et une pour les balises de tâche) et que tous les conteneurs sidecar de la tâche peuvent effectuer ces appels en votre nom. Les appels fréquents aux points de terminaison peuvent entraîner une limitation de l'API.
> Envisagez de mettre en œuvre des stratégies de mise en cache ou de traitement par lots pour réduire la fréquence des appels, en particulier dans les applications à fort trafic, et de résoudre les problèmes de limitation des API en utilisant AWS CloudTrail. Pour plus d'informations sur les seuils de limitation de l'API `ListTagsForResource`, consultez la section Limitation des demandes pour l'API Amazon ECS dans la Référence API Amazon Elastic Container Service. Pour plus d'informations sur le débogage des appels d'API Amazon ECS à l'aide de AWS CloudTrail, voir Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Ce chemin renvoie les statistiques Docker pour ce conteneur spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

Pour les tâches Amazon ECS qui utilisent les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Linux Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur, il y aura des statistiques de débit réseau supplémentaires incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs associés à la tâche. Il peut être utilisé par les conteneurs sidecar pour extraire les métriques réseau. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

Pour les tâches Amazon ECS qui utilisent les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Linux Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur, il y aura des statistiques de débit réseau supplémentaires incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

Réponse JSON des métadonnées de tâches Amazon ECS v4

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI_V4}/task`). Cela inclut les métadonnées associées à la tâche en plus des métadonnées de chaque conteneur dans la tâche.

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`ServiceName`

Nom du service auquel appartient la tâche. ServiceName apparaîtra pour les instances de conteneurs Amazon EC2 et Amazon ECS Anywhere si la tâche est associée à un service.

> **Note**
>
> Les métadonnées `ServiceName` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`VPCID`

L'identifiant VPC de l'instance de conteneur Amazon EC2. Ce champ apparaît uniquement pour les instances Amazon EC2.

> **Note**
>
> Les métadonnées `VPCID` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme (Windows).

`LaunchType`

Type de lancement utilisé par la tâche. Lorsque vous utilisez des fournisseurs de capacité de cluster, cela indique si la tâche utilise l'infrastructure Fargate ou EC2.

> **Note**
>
> Ces métadonnées `LaunchType` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure (Linux) ou 1.0.0 ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

Lorsque vous utilisez Fargate, l'ID est un hexadécimal de 32 chiffres suivi d'un nombre de 10 chiffres.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`LogDriver`

Pilote de journal utilisé par le conteneur.

> **Note**
>
> Ces métadonnées `LogDriver` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`LogOptions`

Options de pilote de journal définies pour le conteneur.

> **Note**
>
> Ces métadonnées `LogOptions` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`ContainerARN`

L'Amazon Resource Name (ARN) du conteneur.

> **Note**
>
> Ces métadonnées `ContainerARN` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`RestartCount`

Nombre de fois où le conteneur a été redémarré.

> **Note**
>
> Les métadonnées `RestartCount` ne sont incluses que si une politique de redémarrage est activée pour le conteneur. Pour plus d'informations, consultez Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v4

Les exemples suivants présentent des sorties de chaque point de terminaison des métadonnées de tâche.

Exemple de réponse de métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de résultat d'une tâche exécutée dans le cadre d'un service (MyService).

```
{
    "DockerId": "ea32192c8553fbff06c9340478a2ff089b2bb5646fb718b4ee206641c9086d66",
    "Name": "curl",
    "DockerName": "ecs-curltest-24-curl-cca48e8dcadd97805600",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/8f03e41243824aea923aca126495f665",
        "com.amazonaws.ecs.task-definition-family": "curltest",
```

```
        "com.amazonaws.ecs.task-definition-version": "24"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-02T00:15:07.620912337Z",
    "StartedAt": "2020-10-02T00:15:08.062559351Z",
    "Type": "NORMAL",
    "LogDriver": "awslogs",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/metadata",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/8f03e41243824aea923aca126495f665"
    },
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/0206b271-
b33f-47ab-86c6-a0ba208a70a9",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.100"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0e:9e:32:c7:48:85",
            "IPv4SubnetCIDRBlock": "10.0.2.0/24",
            "PrivateDNSName": "ip-10-0-2-100.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "10.0.2.1/24"
        }
    ]
}
```

Exemple de réponse de métadonnées de tâches

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie en plus des métadonnées pour chaque conteneur dans la tâche vous sont renvoyées. Voici un exemple de sortie.

```
{
    "Cluster": "default",
```

```
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
```

```
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
```

```
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemple de tâche avec réponse de métadonnées de balises

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, les métadonnées relatives à la tâche, y compris les balises de tâche et d'instance de conteneur vous sont renvoyées. Voici un exemple de sortie.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "TaskTags": {
        "tag-use": "task-metadata-endpoint-test"
    },
    "ContainerInstanceTags":{
        "tag_key":"tag_value"
    },
    "LaunchType": "EC2",
    "Containers": [
        {
```

```
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
```

```
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
```

```
}
```

Exemple de tâche avec balises contenant une réponse de métadonnées d'erreur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`, les métadonnées relatives à la tâche, y compris les balises de tâche et d'instance de conteneur vous sont renvoyées. S'il y a une erreur lors de la récupération des données d'étiquette, l'erreur est renvoyée dans la réponse. Ce qui suit est un exemple de sortie lorsque le rôle IAM associé à l'instance de conteneur n'a pas l'autorisation `ecs:ListTagsForResource`.

```
{
    "Cluster": "default",
    "TaskARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
    "Family": "curltest",
    "ServiceName": "MyService",
    "Revision": "26",
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "PullStartedAt": "2020-10-02T00:43:06.202617438Z",
    "PullStoppedAt": "2020-10-02T00:43:06.31288465Z",
    "AvailabilityZone": "us-west-2d",
    "VPCID": "vpc-1234567890abcdef0",
    "Errors": [
        {
            "ErrorField": "ContainerInstanceTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:container-
instance/default/2dd1b186f39845a584488d2ef155c131",
            "StatusCode": 400,
            "RequestId": "cd597ef0-272b-4643-9bd2-1de469870fa6",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:container-instance/
default/2dd1b186f39845a584488d2ef155c131"
        },
        {
            "ErrorField": "TaskTags",
            "ErrorCode": "AccessDeniedException",
            "ErrorMessage": "User: arn:aws:sts::111122223333:assumed-
role/ecsInstanceRole/i-0744a608689EXAMPLE is not authorized to perform:
 ecs:ListTagsForResource on resource: arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3",
```

```
            "StatusCode": 400,
            "RequestId": "862c5986-6cd2-4aa6-87cc-70be395531e1",
            "ResourceARN": "arn:aws:ecs:us-west-2:111122223333:task/
default/9ef30e4b7aa44d0db562749cff4983f3"
        }
    ],
    "LaunchType": "EC2",
    "Containers": [
        {
            "DockerId":
 "598cba581fe3f939459eaba1e071d5c93bb2c49b7d1ba7db6bb19deeb70d8e38",
            "Name": "~internal~ecs~pause",
            "DockerName": "ecs-curltest-26-internalecspause-e292d586b6f9dade4a00",
            "Image": "amazon/amazon-ecs-pause:0.1.0",
            "ImageID": "",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RESOURCES_PROVISIONED",
            "KnownStatus": "RESOURCES_PROVISIONED",
            "Limits": {
                "CPU": 0,
                "Memory": 0
            },
            "CreatedAt": "2020-10-02T00:43:05.602352471Z",
            "StartedAt": "2020-10-02T00:43:06.076707576Z",
            "Type": "CNI_PAUSE",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
```

```
            ]
        },
        {
            "DockerId":
 "ee08638adaaf009d78c248913f629e38299471d45fe7dc944d1039077e3424ca",
            "Name": "curl",
            "DockerName": "ecs-curltest-26-curl-a0e7dba5aca6d8cb2e00",
            "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
            "ImageID":
 "sha256:d691691e9652791a60114e67b365688d20d19940dde7c4736ea30e660d8d3553",
            "Labels": {
                "com.amazonaws.ecs.cluster": "default",
                "com.amazonaws.ecs.container-name": "curl",
                "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/
default/158d1c8083dd49d6b527399fd6414f5c",
                "com.amazonaws.ecs.task-definition-family": "curltest",
                "com.amazonaws.ecs.task-definition-version": "26"
            },
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Limits": {
                "CPU": 10,
                "Memory": 128
            },
            "CreatedAt": "2020-10-02T00:43:06.326590752Z",
            "StartedAt": "2020-10-02T00:43:06.767535449Z",
            "Type": "NORMAL",
            "LogDriver": "awslogs",
            "LogOptions": {
                "awslogs-create-group": "true",
                "awslogs-group": "/ecs/metadata",
                "awslogs-region": "us-west-2",
                "awslogs-stream": "ecs/curl/158d1c8083dd49d6b527399fd6414f5c"
            },
            "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/
abb51bdd-11b4-467f-8f6c-adcfe1fe059d",
            "Networks": [
                {
                    "NetworkMode": "awsvpc",
                    "IPv4Addresses": [
                        "10.0.2.61"
                    ],
                    "AttachmentIndex": 0,
                    "MACAddress": "0e:10:e2:01:bd:91",
```

```
                    "IPv4SubnetCIDRBlock": "10.0.2.0/24",
                    "PrivateDNSName": "ip-10-0-2-61.us-west-2.compute.internal",
                    "SubnetGatewayIpv4Address": "10.0.2.1/24"
                }
            ]
        }
    ]
}
```

Exemple de réponse des statistiques de conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/stats`, les métriques réseau pour conteneur sont renvoyées. Pour les tâches Amazon ECS qui utilisent les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur, il y aura des statistiques de débit réseau supplémentaires incluses dans la réponse. Pour toutes les autres tâches, la réponse inclura uniquement les statistiques cumulatives du réseau.

Voici un exemple de sortie d'une tâche Amazon ECS sur Amazon EC2 qui utilise le mode réseau `bridge`.

```
{
    "read": "2020-10-02T00:51:13.410254284Z",
    "preread": "2020-10-02T00:51:12.406202398Z",
    "pids_stats": {
        "current": 3
    },
    "blkio_stats": {
        "io_service_bytes_recursive": [

        ],
        "io_serviced_recursive": [

        ],
        "io_queue_recursive": [

        ],
        "io_service_time_recursive": [

        ],
        "io_wait_time_recursive": [
```

```
        ],
        "io_merged_recursive": [

        ],
        "io_time_recursive": [

        ],
        "sectors_recursive": [

        ]
    },
    "num_procs": 0,
    "storage_stats": {

    },
    "cpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13939680000000,
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "precpu_stats": {
        "cpu_usage": {
            "total_usage": 360968065,
            "percpu_usage": [
                182359190,
                178608875
            ],
            "usage_in_kernelmode": 40000000,
            "usage_in_usermode": 290000000
        },
        "system_cpu_usage": 13937670000000,
```

```
        "online_cpus": 2,
        "throttling_data": {
            "periods": 0,
            "throttled_periods": 0,
            "throttled_time": 0
        }
    },
    "memory_stats": {
        "usage": 1806336,
        "max_usage": 6299648,
        "stats": {
            "active_anon": 606208,
            "active_file": 0,
            "cache": 0,
            "dirty": 0,
            "hierarchical_memory_limit": 134217728,
            "hierarchical_memsw_limit": 268435456,
            "inactive_anon": 0,
            "inactive_file": 0,
            "mapped_file": 0,
            "pgfault": 4185,
            "pgmajfault": 0,
            "pgpgin": 2926,
            "pgpgout": 2778,
            "rss": 606208,
            "rss_huge": 0,
            "total_active_anon": 606208,
            "total_active_file": 0,
            "total_cache": 0,
            "total_dirty": 0,
            "total_inactive_anon": 0,
            "total_inactive_file": 0,
            "total_mapped_file": 0,
            "total_pgfault": 4185,
            "total_pgmajfault": 0,
            "total_pgpgin": 2926,
            "total_pgpgout": 2778,
            "total_rss": 606208,
            "total_rss_huge": 0,
            "total_unevictable": 0,
            "total_writeback": 0,
            "unevictable": 0,
            "writeback": 0
        },
```

```
        "limit": 134217728
    },
    "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
    "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
    "networks": {
        "eth0": {
            "rx_bytes": 84,
            "rx_packets": 2,
            "rx_errors": 0,
            "rx_dropped": 0,
            "tx_bytes": 84,
            "tx_packets": 2,
            "tx_errors": 0,
            "tx_dropped": 0
        }
    },
    "network_rate_stats": {
        "rx_bytes_per_sec": 0,
        "tx_bytes_per_sec": 0
    }
}
```

Exemple de réponse des statistiques de tâche

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, les métriques réseau dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
    "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854": {
        "read": "2020-10-02T00:51:32.51467703Z",
        "preread": "2020-10-02T00:51:31.50860463Z",
        "pids_stats": {
            "current": 1
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [
```

```
            ],
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13977820000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 177232665,
                "percpu_usage": [
```

```
                    13376224,
                    163856441
                ],
                "usage_in_kernelmode": 0,
                "usage_in_usermode": 160000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 532480,
            "max_usage": 6279168,
            "stats": {
                "active_anon": 40960,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 9223372036854771712,
                "hierarchical_memsw_limit": 9223372036854771712,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 2033,
                "pgmajfault": 0,
                "pgpgin": 1734,
                "pgpgout": 1724,
                "rss": 40960,
                "rss_huge": 0,
                "total_active_anon": 40960,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 2033,
                "total_pgmajfault": 0,
                "total_pgpgin": 1734,
                "total_pgpgout": 1724,
```

```
                "total_rss": 40960,
                "total_rss_huge": 0,
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 4073377792
        },
        "name": "/ecs-curltest-26-internalecspause-a6bcc3dbadfacfe85300",
        "id": "01999f2e5c6cf4df3873f28950e6278813408f281c54778efec860d0caad4854",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    },
    "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af": {
        "read": "2020-10-02T00:51:32.512771349Z",
        "preread": "2020-10-02T00:51:31.510597736Z",
        "pids_stats": {
            "current": 3
        },
        "blkio_stats": {
            "io_service_bytes_recursive": [

            ],
            "io_serviced_recursive": [

            ],
            "io_queue_recursive": [

            ],
```

```
            "io_service_time_recursive": [

            ],
            "io_wait_time_recursive": [

            ],
            "io_merged_recursive": [

            ],
            "io_time_recursive": [

            ],
            "sectors_recursive": [

            ]
        },
        "num_procs": 0,
        "storage_stats": {

        },
        "cpu_stats": {
            "cpu_usage": {
                "total_usage": 379075681,
                "percpu_usage": [
                    191355275,
                    187720406
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13977800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "precpu_stats": {
            "cpu_usage": {
                "total_usage": 378825197,
                "percpu_usage": [
                    191104791,
                    187720406
```

```
                ],
                "usage_in_kernelmode": 60000000,
                "usage_in_usermode": 310000000
            },
            "system_cpu_usage": 13975800000000,
            "online_cpus": 2,
            "throttling_data": {
                "periods": 0,
                "throttled_periods": 0,
                "throttled_time": 0
            }
        },
        "memory_stats": {
            "usage": 1814528,
            "max_usage": 6299648,
            "stats": {
                "active_anon": 606208,
                "active_file": 0,
                "cache": 0,
                "dirty": 0,
                "hierarchical_memory_limit": 134217728,
                "hierarchical_memsw_limit": 268435456,
                "inactive_anon": 0,
                "inactive_file": 0,
                "mapped_file": 0,
                "pgfault": 5377,
                "pgmajfault": 0,
                "pgpgin": 3613,
                "pgpgout": 3465,
                "rss": 606208,
                "rss_huge": 0,
                "total_active_anon": 606208,
                "total_active_file": 0,
                "total_cache": 0,
                "total_dirty": 0,
                "total_inactive_anon": 0,
                "total_inactive_file": 0,
                "total_mapped_file": 0,
                "total_pgfault": 5377,
                "total_pgmajfault": 0,
                "total_pgpgin": 3613,
                "total_pgpgout": 3465,
                "total_rss": 606208,
                "total_rss_huge": 0,
```

```
                "total_unevictable": 0,
                "total_writeback": 0,
                "unevictable": 0,
                "writeback": 0
            },
            "limit": 134217728
        },
        "name": "/ecs-curltest-26-curl-c2e5f6e0cf91b0bead01",
        "id": "5fc21e5b015f899d22618f8aede80b6d70d71b2a75465ea49d9462c8f3d2d3af",
        "networks": {
            "eth0": {
                "rx_bytes": 84,
                "rx_packets": 2,
                "rx_errors": 0,
                "rx_dropped": 0,
                "tx_bytes": 84,
                "tx_packets": 2,
                "tx_errors": 0,
                "tx_dropped": 0
            }
        },
        "network_rate_stats": {
            "rx_bytes_per_sec": 0,
            "tx_bytes_per_sec": 0
        }
    }
}
```


Point de terminaison des métadonnées de tâches Amazon ECS version 3

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 3 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour plus d'informations, consultez the section called "Point de terminaison des métadonnées de tâches version 4".
>
> Si vous utilisez des tâches Amazon ECS hébergées sur AWS Fargate, voir Point de terminaison de métadonnées de tâches Amazon ECS version 3 pour les tâches sur Fargate.

À partir de la version 1.21.0 de l'agent de conteneur Amazon ECS, l'agent injecte une variable d'environnement appelée `ECS_CONTAINER_METADATA_URI` dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 3, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches. Pour les tâches qui utilisent le mode réseau `bridge`, les métriques réseau sont disponibles sont disponibles lors de l'interrogation des points de terminaison `/stats`.

La fonctionnalité de point de terminaison des métadonnées de tâches version 3 est activée par défaut pour les tâches qui utilisent Fargate sur la version de plateforme v1.3.0 ou ultérieure et les tâches qui utilisent EC2 et qui sont lancées sur l'infrastructure Linux Amazon EC2 exécutant au moins la version 1.21.0 de l'agent de conteneur Amazon ECS ou sur une infrastructure Windows Amazon EC2 exécutant au moins la version `1.54.0` de l'agent de conteneur Amazon ECS et qui utilisent le mode réseau `awsvpc`. Pour plus d'informations, consultez Gestion des instances de conteneur Amazon ECS Linux.

Vous pouvez ajouter la prise en charge de cette fonctionnalité sur des anciennes instances de conteneur en mettant à jour l'agent à l'aide de la dernière version. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

> **Important**
>
> Pour les tâches qui utilisent Fargate et les versions de plateforme antérieures à v1.3.0, le point de terminaison des métadonnées de tâches version 2 est pris en charge. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 2.

Chemins des points de terminaison des métadonnées de tâches version 3

Les points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI}`

Ce chemin renvoie des métadonnées JSON pour le conteneur.

`${ECS_CONTAINER_METADATA_URI}/task`

Ce chemin renvoie le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la

réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâches Amazon ECS v3.

`${ECS_CONTAINER_METADATA_URI}/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API `ListTagsForResource`.

`${ECS_CONTAINER_METADATA_URI}/stats`

Ce chemin renvoie le JSON des statistiques Docker pour cet ID de conteneur Docker spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Ce chemin renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

Réponse JSON des métadonnées de tâches Amazon ECS v3

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise le Service de synchronisation temporelle d’Amazon pour mesurer la précision de l’horloge et fournir l’erreur d’horloge liée aux conteneurs. Pour plus d’informations, consultez la section Réglage de l’heure pour votre instance Linux dans le Guide de l’utilisateur Amazon EC2 pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v3

Les exemples suivants présentent des sorties de points de terminaison des métadonnées de tâche.

Exemple de réponse aux métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de sortie.


```
{
    "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
    "Name": "nginx-curl",
    "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
    "Image": "nrdlngr/nginx-curl",
    "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
    "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 512,
        "Memory": 512
    },
    "CreatedAt": "2018-02-01T20:55:10.554941919Z",
    "StartedAt": "2018-02-01T20:55:11.064236631Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "10.0.2.106"
            ]
        }
    ]
}
```

Exemple de réponse de métadonnées de tâches

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

La réponse JSON suivante est pour une tâche comportant un seul conteneur.


```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
```

```
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
```

```
}
```

Point de terminaison des métadonnées de tâches Amazon ECS version 2

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 2 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour plus d'informations, consultez the section called "Point de terminaison des métadonnées de tâches version 4".

À partir de la version 1.17.0 de l'agent de conteneur Amazon ECS, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS.

Tous les conteneurs appartenant à des tâches qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur Amazon ECS peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Activation des métadonnées de tâche

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 2 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point de terminaison des métadonnées. Pour plus d'informations, consultez the section called "Point de terminaison des métadonnées de tâches version 4".

À partir de la version 1.17.0 de l'agent de conteneur Amazon ECS, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS.

Tous les conteneurs appartenant à des tâches qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur Amazon ECS peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Activation des métadonnées de tâche

La fonctionnalité des métadonnées de tâches version 2 est activée par défaut pour les éléments suivants :

- Tâches utilisant Fargate qui ont recours à la version de plateforme v1.1.0 ou ultérieure. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.
- Tâches utilisant EC2 qui utilisent également le mode réseau `awsvpc` et qui sont lancées sur l'infrastructure Linux Amazon EC2 exécutant au moins la version 1.17.0 de l'agent de conteneur Amazon ECS ou sur l'infrastructure Windows Amazon EC2 exécutant au moins la version `1.54.0` de l'agent du conteneur Amazon ECS. Pour plus d'informations, consultez Gestion des instances de conteneur Amazon ECS Linux.

Vous pouvez ajouter la prise en charge de cette fonctionnalité sur des anciennes instances de conteneur en mettant à jour l'agent à l'aide de la dernière version. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

Chemins de points de terminaison des métadonnées de tâches

Les points de terminaison d'API suivants sont disponibles pour les conteneurs :

`169.254.170.2/v2/metadata`

Ce point de terminaison renvoie le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâche.

`169.254.170.2/v2/metadata/<container-id>`

Ce point de terminaison renvoie le JSON des métadonnées pour l'ID de conteneur Docker spécifié.

`169.254.170.2/v2/metadata/taskWithTags`

Ce chemin renvoie les métadonnées de la tâche incluse dans le point de terminaison `/task` en plus des balises de tâche et d'instance de conteneur qui peuvent être récupérées à l'aide de l'API `ListTagsForResource`.

`169.254.170.2/v2/stats`

Ce point de terminaison renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

`169.254.170.2/v2/stats/<container-id>`

Ce point de terminaison renvoie le JSON des statistiques Docker pour l'ID de conteneur Docker spécifié. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

Réponse JSON des métadonnées de tâche

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`169.254.170.2/v2/metadata`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise le Service de synchronisation temporelle d’Amazon pour mesurer la précision de l’horloge et fournir l’erreur d’horloge liée aux conteneurs. Pour plus d’informations, consultez la section Réglage de l’heure pour votre instance Linux dans le Guide de l’utilisateur Amazon EC2 pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemple de réponse de métadonnées de tâches

La réponse JSON suivante est pour une tâche comportant un seul conteneur.


```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
```

```
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```

Métadonnées de tâches Amazon ECS disponibles pour les tâches sur Fargate

Amazon ECS sur Fargate fournit une méthode pour récupérer diverses métadonnées, métriques réseau et statistiques Docker relatives à vos conteneurs et aux tâches dont ils font partie. On parle alors du point de terminaison des métadonnées de tâches. Les versions de point de terminaison de métadonnées de tâche suivantes sont disponibles pour les tâches Amazon ECS sur Fargate :

- Point de terminaison des métadonnées de tâches version 4 – Disponible pour les tâches qui utilisent la version de plateforme 1.4.0 ou ultérieure.
- Point de terminaison des métadonnées de tâches version 3 – Disponible pour les tâches qui utilisent la version de plateforme 1.1.0 ou ultérieure.

Tous les conteneurs appartenant à des tâches qui sont lancées avec le mode réseau `awsvpc` reçoivent une adresse IPv4 locale au sein d'une plage d'adresses link-local prédéfinie. Lorsqu'un conteneur interroge le point de terminaison des métadonnées, l'agent de conteneur peut déterminer à quelles tâches le conteneur appartient en fonction de son adresse IP unique. Des métadonnées et des statistiques pour cette tâche sont alors renvoyées.

Rubriques

- Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate
- Point de terminaison des métadonnées de tâches Amazon ECS version 3 pour les tâches sur Fargate

Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate

> **Important**
>
> Si vous utilisez des tâches Amazon ECS hébergées sur des instances Amazon EC2, consultez la section Point de terminaison des métadonnées de tâche Amazon ECS.

À partir de la version `1.4.0` de la plateforme Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI_V4` est injectée dans chaque conteneur d'une tâche. Lorsque

vous interrogez le point de terminaison des métadonnées de tâches version 4, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches.

Le point de terminaison des métadonnées de tâche version 4 fonctionne comme dans la version 3, mais fournit des métadonnées réseau supplémentaires pour vos conteneurs et tâches. Des métriques réseau supplémentaires sont disponibles lors de l'interrogation des points de terminaison `/stats`.

Le point de terminaison des métadonnées des tâches est activé par défaut pour toutes les tâches Amazon ECS exécutées sur AWS Fargate qui utilisent la version de la plateforme `1.4.0` ou plus tard.

> **Note**
>
> Pour éviter de créer d'autres versions de point de terminaison de métadonnées de tâche à l'avenir, des métadonnées supplémentaires peuvent être ajoutées à la sortie de la version 4. Nous ne supprimons pas les métadonnées existantes et ne modifions pas les noms des champs de métadonnées.

Chemins des points de terminaison des métadonnées de tâches Fargate version 4

Les points de terminaison des métadonnées de tâches suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI_V4}`

Ce chemin renvoie des métadonnées pour le conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task`

Ce chemin renvoie les métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate.

`${ECS_CONTAINER_METADATA_URI_V4}/stats`

Ce chemin renvoie les statistiques Docker pour le conteneur Docker. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

> **Note**
>
> Tâches Amazon ECS sur AWS Fargate exiger que le conteneur s'exécute pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

`${ECS_CONTAINER_METADATA_URI_V4}/task/stats`

Ce chemin renvoie les statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

> **Note**
>
> Tâches Amazon ECS sur AWS Fargate exiger que le conteneur s'exécute pendant environ 1 seconde avant de renvoyer les statistiques du conteneur.

Réponse JSON des métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate

Les métadonnées suivantes sont renvoyées dans la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI_V4}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`ServiceName`

Nom du service auquel appartient la tâche. ServiceName apparaîtra si les tâches Fargate sont associées à un service.

`VPCID`

L'identifiant VPC de l'instance de conteneur Amazon EC2. Ce champ apparaît uniquement pour les instances Amazon EC2.

> **Note**
>
> Les métadonnées `VPCID` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS `1.63.1` ou version ultérieure.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme (Windows).

`LaunchType`

Type de lancement utilisé par la tâche. Lorsque vous utilisez des fournisseurs de capacité de cluster, cela indique si la tâche utilise l'infrastructure Fargate ou EC2.

> **Note**
>
> Ces métadonnées `LaunchType` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure (Linux) ou 1.0.0 ou ultérieure (Windows).

`EphemeralStorageMetrics`

La taille réservée et l'utilisation actuelle du stockage éphémère de cette tâche.

> **Note**
>
> Fargate réserve de l'espace sur le disque. Il n'est utilisé que par Fargate. Vous n'êtes pas facturé pour cela. Cela n'apparaît pas dans ces statistiques. Toutefois, vous pouvez voir ce stockage supplémentaire dans d'autres outils tels que `df`.

`Utilized`

La quantité de stockage éphémère actuellement utilisée (en Mio) pour cette tâche.

`Reserved`

La quantité de stockage éphémère réservée (en Mio) pour cette tâche. La taille de stockage éphémère ne peut pas être modifiée dans une tâche en cours d'exécution. Vous pouvez spécifier l'objet du `ephermalStorage` dans la définition de votre tâche pour modifier la

quantité de stockage éphémère. Le `ephermalStorage` est spécifié en Gio, et non en Mio. Le `ephermalStorage` et les `EphemeralStorageMetrics` sont uniquement disponibles sur la plateforme Fargate Linux version 1.4.0 ou ultérieure.

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

Lorsque vous utilisez Fargate, l'ID est un hexadécimal de 32 chiffres suivi d'un nombre de 10 chiffres.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`LogDriver`

Pilote de journal utilisé par le conteneur.

> **Note**
>
> Ces métadonnées `LogDriver` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`LogOptions`

Options de pilote de journal définies pour le conteneur.

> **Note**
>
> Ces métadonnées `LogOptions` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`ContainerARN`

L'Amazon Resource Name (ARN) du conteneur.

> **Note**
>
> Ces métadonnées `ContainerARN` sont incluses uniquement lors de l'utilisation de la version de l'agent de conteneur Amazon ECS Linux `1.45.0` ou version ultérieure.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`Snapshotter`

Le snapshotter qui a été utilisé par containerd pour télécharger cette image de conteneur. Les valeurs valides sont `overlayfs`, qui est la valeur par défaut, et `soci`, utilisées en cas de chargement différé avec un index SOCI. Ce paramètre est uniquement disponible pour les tâches qui s'exécutent sur une version de plateforme Linux `1.4.0`.

`RestartCount`

Nombre de fois où le conteneur a été redémarré.

> **Note**
>
> Les métadonnées `RestartCount` ne sont incluses que si une politique de redémarrage est activée pour le conteneur. Pour plus d'informations, consultez

Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cette fonctionnalité utilise le Service de synchronisation temporelle d'Amazon pour mesurer la précision de l'horloge et fournir l'erreur d'horloge liée aux conteneurs. Pour plus d'informations, consultez la section Réglage de l'heure pour votre instance Linux dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate

Les exemples suivants présentent des sorties de points de terminaison des métadonnées de tâche pour les tâches Amazon ECS exécutées sur AWS Fargate.

Depuis le conteneur, vous pouvez utiliser curl suivi du point de terminaison des métadonnées de la tâche pour interroger le point de terminaison, par exemple `curl ${ECS_CONTAINER_METADATA_URI_V4}/task`.

Exemple de réponse de métadonnées du conteneur

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}`, seules les métadonnées sur le conteneur lui-même vous sont renvoyées. Voici un exemple de sortie.


```
{
    "DockerId": "cd189a933e5849daa93386466019ab50-2495160603",
    "Name": "curl",
    "DockerName": "curl",
    "Image": "111122223333.dkr.ecr.us-west-2.amazonaws.com/curltest:latest",
    "ImageID":
 "sha256:25f3695bedfb454a50f12d127839a68ad3caf91e451c1da073db34c542c4d2cb",
    "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-west-2:111122223333:cluster/
default",
        "com.amazonaws.ecs.container-name": "curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-west-2:111122223333:task/default/
cd189a933e5849daa93386466019ab50",
        "com.amazonaws.ecs.task-definition-family": "curltest",
        "com.amazonaws.ecs.task-definition-version": "2"
    },
    "DesiredStatus": "RUNNING",
    "KnownStatus": "RUNNING",
    "Limits": {
        "CPU": 10,
        "Memory": 128
    },
    "CreatedAt": "2020-10-08T20:09:11.44527186Z",
    "StartedAt": "2020-10-08T20:09:11.44527186Z",
    "Type": "NORMAL",
    "Networks": [
        {
            "NetworkMode": "awsvpc",
            "IPv4Addresses": [
                "192.0.2.3"
            ],
            "AttachmentIndex": 0,
            "MACAddress": "0a:de:f6:10:51:e5",
            "IPv4SubnetCIDRBlock": "192.0.2.0/24",
            "DomainNameServers": [
                "192.0.2.2"
            ],
            "DomainNameSearchList": [
                "us-west-2.compute.internal"
```

```
            ],
            "PrivateDNSName": "ip-10-0-0-222.us-west-2.compute.internal",
            "SubnetGatewayIpv4Address": "192.0.2.0/24"
        }
    ],
    "ContainerARN": "arn:aws:ecs:us-west-2:111122223333:container/05966557-
f16c-49cb-9352-24b3a0dcd0e1",
    "LogOptions": {
        "awslogs-create-group": "true",
        "awslogs-group": "/ecs/containerlogs",
        "awslogs-region": "us-west-2",
        "awslogs-stream": "ecs/curl/cd189a933e5849daa93386466019ab50"
    },
    "LogDriver": "awslogs",
    "Snapshotter": "overlayfs"
}
```

Exemples de métadonnées de tâches Amazon ECS v4 pour les tâches sur Fargate

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task`, les métadonnées relatives à la tâche dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.

```
{
  "Cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/MyEmptyCluster",
  "TaskARN": "arn:aws:ecs:us-east-1:123456789012:task/MyEmptyCluster/
bfa2636268144d039771334145e490c5",
  "Family": "sample-fargate",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Limits": {
    "CPU": 0.25,
    "Memory": 512
  },
  "PullStartedAt": "2023-07-21T15:45:33.532811081Z",
  "PullStoppedAt": "2023-07-21T15:45:38.541068435Z",
  "AvailabilityZone": "us-east-1d",
  "Containers": [
    {
      "DockerId": "bfa2636268144d039771334145e490c5-1117626119",
      "Name": "curl-image",
      "DockerName": "curl-image",
```

```
      "Image": "curlimages/curl",
      "ImageID":
 "sha256:daf3f46a2639c1613b25e85c9ee4193af8a1d538f92483d67f9a3d7f21721827",
      "Labels": {
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "curl-image",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 128 },
      "CreatedAt": "2023-07-21T15:45:44.91368314Z",
      "StartedAt": "2023-07-21T15:45:44.91368314Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/da6cccf7-1178-400c-afdf-7536173ee209",
      "Snapshotter": "overlayfs"
    },
    {
      "DockerId": "bfa2636268144d039771334145e490c5-3681984407",
      "Name": "fargate-app",
      "DockerName": "fargate-app",
      "Image": "public.ecr.aws/docker/library/httpd:latest",
      "ImageID":
 "sha256:8059bdd0058510c03ae4c808de8c4fd2c1f3c1b6d9ea75487f1e5caa5ececa02",
      "Labels": {
```

```
        "com.amazonaws.ecs.cluster": "arn:aws:ecs:us-east-1:123456789012:cluster/
MyEmptyCluster",
        "com.amazonaws.ecs.container-name": "fargate-app",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-east-1:123456789012:task/
MyEmptyCluster/bfa2636268144d039771334145e490c5",
        "com.amazonaws.ecs.task-definition-family": "sample-fargate",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": { "CPU": 2 },
      "CreatedAt": "2023-07-21T15:45:44.954460255Z",
      "StartedAt": "2023-07-21T15:45:44.954460255Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": ["172.31.42.189"],
          "AttachmentIndex": 0,
          "MACAddress": "0e:98:9f:33:76:d3",
          "IPv4SubnetCIDRBlock": "172.31.32.0/20",
          "DomainNameServers": ["172.31.0.2"],
          "DomainNameSearchList": ["ec2.internal"],
          "PrivateDNSName": "ip-172-31-42-189.ec2.internal",
          "SubnetGatewayIpv4Address": "172.31.32.1/20"
        }
      ],
      "ContainerARN": "arn:aws:ecs:us-east-1:123456789012:container/MyEmptyCluster/
bfa2636268144d039771334145e490c5/f65b461d-aa09-4acb-a579-9785c0530cbc",
      "Snapshotter": "overlayfs"
    }
  ],
  "LaunchType": "FARGATE",
  "ClockDrift": {
    "ClockErrorBound": 0.446931,
    "ReferenceTimestamp": "2023-07-21T16:09:17Z",
    "ClockSynchronizationStatus": "SYNCHRONIZED"
  },
  "EphemeralStorageMetrics": {
    "Utilized": 261,
    "Reserved": 20496
  }
}
```

Exemple de réponse des statistiques de tâche

Lors de l'interrogation du point de terminaison `${ECS_CONTAINER_METADATA_URI_V4}/task/stats`, les métriques réseau dont le conteneur fait partie vous sont renvoyées. Voici un exemple de sortie.


```
{
  "3d1f891cded94dc795608466cce8ddcf-464223573": {
    "read": "2020-10-08T21:24:44.938937019Z",
    "preread": "2020-10-08T21:24:34.938633969Z",
    "pids_stats": {},
    "blkio_stats": {
      "io_service_bytes_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 638976
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 638976
        }
```

```
      ],
      "io_serviced_recursive": [
        {
          "major": 202,
          "minor": 26368,
          "op": "Read",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Write",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Sync",
          "value": 12
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Async",
          "value": 0
        },
        {
          "major": 202,
          "minor": 26368,
          "op": "Total",
          "value": 12
        }
      ],
      "io_queue_recursive": [],
      "io_service_time_recursive": [],
      "io_wait_time_recursive": [],
      "io_merged_recursive": [],
      "io_time_recursive": [],
      "sectors_recursive": []
    },
    "num_procs": 0,
    "storage_stats": {},
    "cpu_stats": {
      "cpu_usage": {
```

```
        "total_usage": 1137691504,
        "percpu_usage": [
          696479228,
          441212276,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9393210000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "precpu_stats": {
      "cpu_usage": {
        "total_usage": 1136624601,
        "percpu_usage": [
          695639662,
          440984939,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
          0,
```

```
          0,
          0,
          0,
          0
        ],
        "usage_in_kernelmode": 80000000,
        "usage_in_usermode": 810000000
      },
      "system_cpu_usage": 9373330000000,
      "online_cpus": 2,
      "throttling_data": {
        "periods": 0,
        "throttled_periods": 0,
        "throttled_time": 0
      }
    },
    "memory_stats": {
      "usage": 6504448,
      "max_usage": 8458240,
      "stats": {
        "active_anon": 1675264,
        "active_file": 557056,
        "cache": 651264,
        "dirty": 0,
        "hierarchical_memory_limit": 536870912,
        "hierarchical_memsw_limit": 9223372036854772000,
        "inactive_anon": 0,
        "inactive_file": 3088384,
        "mapped_file": 430080,
        "pgfault": 11034,
        "pgmajfault": 5,
        "pgpgin": 8436,
        "pgpgout": 7137,
        "rss": 4669440,
        "rss_huge": 0,
        "total_active_anon": 1675264,
        "total_active_file": 557056,
        "total_cache": 651264,
        "total_dirty": 0,
        "total_inactive_anon": 0,
        "total_inactive_file": 3088384,
        "total_mapped_file": 430080,
        "total_pgfault": 11034,
        "total_pgmajfault": 5,
```

```
        "total_pgpgin": 8436,
        "total_pgpgout": 7137,
        "total_rss": 4669440,
        "total_rss_huge": 0,
        "total_unevictable": 0,
        "total_writeback": 0,
        "unevictable": 0,
        "writeback": 0
      },
      "limit": 9223372036854772000
    },
    "name": "curltest",
    "id": "3d1f891cded94dc795608466cce8ddcf-464223573",
    "networks": {
      "eth1": {
        "rx_bytes": 2398415937,
        "rx_packets": 1898631,
        "rx_errors": 0,
        "rx_dropped": 0,
        "tx_bytes": 1259037719,
        "tx_packets": 428002,
        "tx_errors": 0,
        "tx_dropped": 0
      }
    },
    "network_rate_stats": {
      "rx_bytes_per_sec": 43.298687872232854,
      "tx_bytes_per_sec": 215.39347269466413
    }
  }
}
```


Point de terminaison des métadonnées de tâches Amazon ECS version 3 pour les tâches sur Fargate

> **Important**
>
> Le point de terminaison des métadonnées de tâches version 3 n'est plus activement maintenu. Nous vous recommandons de mettre à jour le point de terminaison des métadonnées de tâches version 4 afin d'obtenir les dernières informations relatives au point

de terminaison des métadonnées. Pour plus d'informations, consultez the section called "Point de terminaison des métadonnées de tâches version 4 pour les tâches sur Fargate".

À partir de la version `1.1.0` de la plateforme Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonnées de tâches version 3, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches.

La fonction de point de terminaison des métadonnées de tâches est activée par défaut pour les tâches Amazon ECS hébergées sur Fargate qui utilisent la version de plateforme `1.1.0` ou ultérieure. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.

Chemins de point de terminaison des métadonnées de tâches sur Fargate

Les points de terminaison d'API suivants sont disponibles pour les conteneurs :

`${ECS_CONTAINER_METADATA_URI}`

Ce chemin renvoie des métadonnées JSON pour le conteneur.

`${ECS_CONTAINER_METADATA_URI}/task`

Ce chemin renvoie le JSON des métadonnées pour la tâche, y compris une liste des ID et noms de conteneur pour tous les conteneurs associés à la tâche. Pour plus d'informations sur la réponse pour ce point de terminaison, consultez Réponse JSON des métadonnées de tâches Amazon ECS v3 pour les tâches sur Fargate.

`${ECS_CONTAINER_METADATA_URI}/stats`

Ce chemin renvoie le JSON des statistiques Docker pour cet ID de conteneur Docker spécifique. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

`${ECS_CONTAINER_METADATA_URI}/task/stats`

Ce chemin renvoie le JSON des statistiques Docker pour tous les conteneurs associés à la tâche. Pour plus d'informations sur chacune des statistiques renvoyées, voir ContainerStats dans la documentation de l'API Docker.

Réponse JSON des métadonnées de tâches Amazon ECS v3 pour les tâches sur Fargate

Les informations suivantes sont renvoyées à partir de la réponse JSON du point de terminaison de métadonnées de la tâche (`${ECS_CONTAINER_METADATA_URI}/task`).

`Cluster`

Amazon Resource Name (ARN) ou nom abrégé du cluster Amazon ECS auquel la tâche appartient.

`TaskARN`

L'Amazon Resource Name (ARN) de la tâche à laquelle le conteneur appartient.

`Family`

Famille de la définition de tâche Amazon ECS pour la tâche.

`Revision`

Révision de la définition de tâche Amazon ECS pour la tâche.

`DesiredStatus`

État souhaité pour la tâche à partir d'Amazon ECS.

`KnownStatus`

État connu pour la tâche à partir d'Amazon ECS.

`Limits`

Limites de ressource spécifiées au niveau de la tâche, par exemple, le processeur (exprimé en vCPU) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`PullStartedAt`

Horodatage au moment où la première extraction de l'image de conteneur a commencé.

`PullStoppedAt`

Horodatage au moment où la dernière extraction de l'image de conteneur s'est terminée.

`AvailabilityZone`

Zone de disponibilité dans laquelle se trouve la tâche.

> **Note**
>
> Les métadonnées de la zone de disponibilité sont uniquement disponibles pour les tâches Fargate qui utilisent la version 1.4 de la plateforme ou ultérieure (Linux) ou la version 1.0.0 de la plateforme ou ultérieure (Windows).

`Containers`

Liste de métadonnées de conteneur pour chaque conteneur associé à la tâche.

`DockerId`

ID Docker du conteneur.

`Name`

Nom du conteneur tel que spécifié dans la définition de tâche.

`DockerName`

Nom du conteneur fourni à Docker. L'agent de conteneur Amazon ECS génère un nom unique pour le conteneur afin d'éviter les conflits de noms lorsque plusieurs copies de la même définition de tâche sont exécutées sur une seule instance.

`Image`

Image à utiliser pour le conteneur.

`ImageID`

Le SHA-256 résumé de l'image manifeste. Il s’agit du condensé qui peut être utilisé pour extraire l’image en utilisant le format `repository-url/image@sha256:digest`.

`Ports`

Tous les ports exposés pour le conteneur. Ce paramètre n'est pas spécifié s'il n'y a pas de ports exposés.

`Labels`

Toutes les étiquettes appliquées au conteneur. Ce paramètre n'est pas spécifié si aucune étiquette n'est appliquée.

`DesiredStatus`

État souhaité pour le conteneur à partir d'Amazon ECS.

`KnownStatus`

État connu pour le conteneur à partir d'Amazon ECS.

`ExitCode`

Code de sortie pour le conteneur. Ce paramètre n'est pas spécifié si le conteneur n'est pas sorti.

`Limits`

Limites de ressource spécifiées au niveau du conteneur, par exemple, le processeur (exprimé en unités de processeur) et la mémoire. Ce paramètre n'est pas spécifié si aucune limite de ressource n'a été définie.

`CreatedAt`

Horodatage de création du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été créé.

`StartedAt`

Horodatage de démarrage du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été démarré.

`FinishedAt`

Horodatage d'arrêt du conteneur. Ce paramètre n'est pas spécifié si le conteneur n'a pas encore été arrêté.

`Type`

Type du conteneur. Les conteneurs qui sont spécifiés dans votre définition de tâche sont de type `NORMAL`. Vous pouvez ignorer les autres types de conteneurs, qui sont utilisés pour l'approvisionnement des ressources de tâches internes par l'agent de conteneur Amazon ECS.

`Networks`

Informations de réseau du conteneur, par exemple le mode réseau et l'adresse IP. Ce paramètre n'est pas spécifié si aucune information de réseau n'est définie.

`ClockDrift`

Informations sur la différence entre l'heure de référence et l'heure du système. Cela s'applique au système d'exploitation Linux. Cette fonctionnalité utilise le Service de synchronisation temporelle d’Amazon pour mesurer la précision de l’horloge et fournir l’erreur d’horloge liée aux conteneurs.

Pour plus d'informations, consultez la section Réglage de l'heure pour votre instance Linux dans le Guide de l'utilisateur Amazon EC2 pour les instances Linux.

`ReferenceTime`

La base de la précision de l'horloge. Amazon ECS utilise la norme mondiale du temps universel coordonné (UTC) via NTP, par exemple `2021-09-07T16:57:44Z`.

`ClockErrorBound`

La mesure de l'erreur de l'horloge est exprimée par le décalage par rapport à l'UTC. Cette erreur correspond à la différence en millisecondes entre l'heure de référence et l'heure du système.

`ClockSynchronizationStatus`

Indique si la tentative de synchronisation la plus récente entre l'heure système et l'heure de référence a réussi.

Les valeurs valides sont `SYNCHRONIZED` et `NOT_SYNCHRONIZED`.

`ExecutionStoppedAt`

Horodatage au moment où l'état des tâches est passé de `DesiredStatus` à `STOPPED`. Cela se produit lorsqu'un conteneur essentiel passe à `STOPPED`.

Exemples de métadonnées de tâches Amazon ECS v3 pour les tâches sur Fargate

La réponse JSON suivante est pour une tâche comportant un seul conteneur.


```
{
  "Cluster": "default",
  "TaskARN": "arn:aws:ecs:us-east-2:012345678910:task/9781c248-0edd-4cdb-9a93-
f63cb662a5d3",
  "Family": "nginx",
  "Revision": "5",
  "DesiredStatus": "RUNNING",
  "KnownStatus": "RUNNING",
  "Containers": [
    {
      "DockerId": "731a0d6a3b4210e2448339bc7015aaa79bfe4fa256384f4102db86ef94cbbc4c",
      "Name": "~internal~ecs~pause",
      "DockerName": "ecs-nginx-5-internalecspause-acc699c0cbf2d6d11700",
      "Image": "amazon/amazon-ecs-pause:0.1.0",
      "ImageID": "",
```

```
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "~internal~ecs~pause",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RESOURCES_PROVISIONED",
      "KnownStatus": "RESOURCES_PROVISIONED",
      "Limits": {
        "CPU": 0,
        "Memory": 0
      },
      "CreatedAt": "2018-02-01T20:55:08.366329616Z",
      "StartedAt": "2018-02-01T20:55:09.058354915Z",
      "Type": "CNI_PAUSE",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    },
    {
      "DockerId": "43481a6ce4842eec8fe72fc28500c6b52edcc0917f105b83379f88cac1ff3946",
      "Name": "nginx-curl",
      "DockerName": "ecs-nginx-5-nginx-curl-ccccb9f49db0dfe0d901",
      "Image": "nrdlngr/nginx-curl",
      "ImageID":
 "sha256:2e00ae64383cfc865ba0a2ba37f61b50a120d2d9378559dcd458dc0de47bc165",
      "Labels": {
        "com.amazonaws.ecs.cluster": "default",
        "com.amazonaws.ecs.container-name": "nginx-curl",
        "com.amazonaws.ecs.task-arn": "arn:aws:ecs:us-
east-2:012345678910:task/9781c248-0edd-4cdb-9a93-f63cb662a5d3",
        "com.amazonaws.ecs.task-definition-family": "nginx",
        "com.amazonaws.ecs.task-definition-version": "5"
      },
      "DesiredStatus": "RUNNING",
      "KnownStatus": "RUNNING",
      "Limits": {
```

```
        "CPU": 512,
        "Memory": 512
      },
      "CreatedAt": "2018-02-01T20:55:10.554941919Z",
      "StartedAt": "2018-02-01T20:55:11.064236631Z",
      "Type": "NORMAL",
      "Networks": [
        {
          "NetworkMode": "awsvpc",
          "IPv4Addresses": [
            "10.0.2.106"
          ]
        }
      ]
    }
  ],
  "PullStartedAt": "2018-02-01T20:55:09.372495529Z",
  "PullStoppedAt": "2018-02-01T20:55:10.552018345Z",
  "AvailabilityZone": "us-east-2b"
}
```


Introspection de conteneur Amazon ECS

L'agent de conteneur Amazon ECS fournit une opération API pour l'obtention d'informations sur l'instance de conteneur sur laquelle l'agent est exécuté et sur les tâches connexes elles aussi exécutées sur cette instance. Vous pouvez utiliser la commande curl à partir de l'instance de conteneur pour interroger l'agent de conteneur Amazon ECS (port 51678) et renvoyer des métadonnées d'instance de conteneur ou des informations sur les tâches.

> **Important**
>
> Votre instance de conteneur doit avoir un rôle IAM qui autorise l'accès à Amazon ECS afin de récupérer les métadonnées. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

Pour afficher les métadonnées de l'instance de conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante. Les métadonnées comprennent l'ID de l'instance de conteneur, le cluster Amazon ECS dans lequel l'instance de conteneur est enregistrée, ainsi que les informations de version de l'agent de conteneur Amazon ECS.

```
curl -s http://localhost:51678/v1/metadata | python3 -mjson.tool
```

Sortie :

```
{
    "Cluster": "cluster_name",
    "ContainerInstanceArn": "arn:aws:ecs:region:aws_account_id:container-
instance/cluster_name/container_instance_id",
    "Version": "Amazon ECS Agent - v1.30.0 (02ff320c)"
}
```

Pour afficher les informations sur toutes les tâches qui sont exécutées sur une instance de conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante :

```
curl http://localhost:51678/v1/tasks
```

Sortie :

```
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/example5-58ff-46c9-
ae05-543f8example",
            "DesiredStatus": "RUNNING",
            "KnownStatus": "RUNNING",
            "Family": "hello_world",
            "Version": "8",
            "Containers": [
                {
          "DockerId":
 "9581a69a761a557fbfce1d0f6745e4af5b9dbfb86b6b2c5c4df156f1a5932ff1",
          "DockerName": "ecs-hello_world-8-mysql-fcae8ac8f9f1d89d8301",
          "Name": "mysql",
          "CreatedAt": "2023-10-08T20:09:11.44527186Z",
          "StartedAt": "2023-10-08T20:09:11.44527186Z",
          "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        },
        {
          "DockerId":
 "bf25c5c5b2d4dba68846c7236e75b6915e1e778d31611e3c6a06831e39814a15",
```

```
            "DockerName": "ecs-hello_world-8-wordpress-e8bfddf9b488dff36c00",
            "Name": "wordpress"
        }
    ]
}
    ]
}
```

Vous pouvez afficher les informations relatives à une tâche spécifique en cours d'exécution sur une instance de conteneur. Pour spécifier une tâche ou un conteneur spécifique, ajoutez un des éléments suivants à la demande :

- La tâche ARN (`?taskarn=`*`task_arn`*)
- L'ID Docker d'un conteneur (`?dockerid=`*`docker_id`*)

Pour obtenir des informations sur une tâche avec l'ID Docker d'un conteneur, connectez-vous à votre instance de conteneur via SSH et exécutez la commande suivante.

> **Note**
>
> Les agents de conteneur Amazon ECS antérieurs à la version 1.14.2 nécessitent des ID de conteneur Docker complets pour l'API d'introspection, et pas seulement la version courte qui est affichée avec la commande docker ps. Vous pouvez obtenir l'ID Docker complet d'un conteneur en exécutant la commande docker ps --no-trunc sur l'instance du conteneur.

```
curl http://localhost:51678/v1/tasks?dockerid=79c796ed2a7f
```

Sortie :

```
{
    "Arn": "arn:aws:ecs:us-west-2:012345678910:task/default/e01d58a8-151b-40e8-
bc01-22647b9ecfec",
    "Containers": [
        {
            "DockerId":
 "79c796ed2a7f864f485c76f83f3165488097279d296a7c05bd5201a1c69b2920",
            "DockerName": "ecs-nginx-efs-2-nginx-9ac0808dd0afa495f001",
            "Name": "nginx",
```

```
            "CreatedAt": "2023-10-08T20:09:11.44527186Z",
            "StartedAt": "2023-10-08T20:09:11.44527186Z",
            "ImageID":
 "sha256:2ae34abc2ed0a22e280d17e13f9c01aaf725688b09b7a1525d1a2750e2c0d1de"
        }
    ],
    "DesiredStatus": "RUNNING",
    "Family": "nginx-efs",
    "KnownStatus": "RUNNING",
    "Version": "2"
}
```

Identification des comportements non autorisés grâce à la surveillance d’exécution

Amazon GuardDuty est un service de détection des menaces qui permet de protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, GuardDuty surveille en permanence les différentes sources de journaux et les activités d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

Surveillance du temps d'exécution dans GuardDuty protège les charges de travail exécutées sur les instances de conteneurs Fargate et EC2 grâce à une surveillance continue AWS journal et activité réseau pour identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un outil léger et entièrement géré GuardDuty agent de sécurité qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l’escalade des privilèges, l’utilisation d’identifiants exposés ou la communication avec des adresses IP et des domaines malveillants, ainsi que la présence de logiciels malveillants sur vos instances Amazon EC2 et vos charges de travail conteneurisées. Pour plus d'informations, voir GuardDuty Surveillance du temps d'exécution dans le GuardDuty Guide de l'utilisateur.

Votre administrateur de sécurité active la surveillance de l'exécution pour un ou plusieurs comptes dans AWS Organizations pour GuardDuty. Ils sélectionnent également si GuardDuty déploie automatiquement le GuardDuty agent de sécurité lorsque vous utilisez Fargate. Tous vos clusters sont automatiquement protégés, et GuardDuty gère l'agent de sécurité en votre nom.

Vous pouvez également configurer manuellement GuardDuty agent de sécurité dans les cas suivants :

- Vous utilisez des instances de conteneur EC2
- Vous avez besoin d'un contrôle granulaire pour activer la surveillance d'exécution au niveau du cluster

Pour utiliser Runtime Monitoring, vous devez configurer les clusters protégés, installer et gérer GuardDuty agent de sécurité sur vos instances de conteneur EC2.

Comment fonctionne la surveillance d'exécution avec Amazon◦ECS

Runtime Monitoring utilise un GuardDuty agent de sécurité qui surveille l'activité de la charge de travail Amazon ECS pour savoir comment les applications demandent, accèdent et consomment les ressources système sous-jacentes.

Pour les tâches Fargate, le GuardDuty L'agent de sécurité fonctionne comme un conteneur annexe pour chaque tâche.

Pour les instances de conteneur EC2, le GuardDuty l'agent de sécurité s'exécute en tant que processus sur l'instance.

Le GuardDuty l'agent de sécurité collecte des données à partir des ressources suivantes, puis les envoie à GuardDuty à traiter. Vous pouvez consulter les résultats dans GuardDuty console. Vous pouvez également les envoyer à d'autres Services AWS tels que AWS Security Hub CSPM, ou un fournisseur de sécurité tiers pour l'agrégation et la correction. Pour plus d'informations sur la façon d'afficher et de gérer les résultats, voir Gestion d'Amazon GuardDuty résultats dans le Amazon GuardDuty Guide de l'utilisateur.

- Réponses provenant des appels API Amazon ECS suivants :
  - DescribeClusters

    Les paramètres de réponse incluent la balise de surveillance d'exécution (lorsque celle-ci est définie) lorsque vous utilisez l'option `--include TAGS`.
  - DescribeTasks

    Pour Fargate, les paramètres de réponse incluent GuardDuty conteneur sidecar.
  - ListAccountSettings

    Les paramètres de réponse incluent le paramètre du compte de surveillance d'exécution, qui est défini par votre administrateur de sécurité.

- Les données d'introspection de l'agent de conteneur. Pour plus d'informations, consultez Introspection de conteneur Amazon ECS.
- Point de terminaison des métadonnées de tâches pour l'option de calcul :
  - Point de terminaison des métadonnées de tâches Amazon ECS version 4
  - Point de terminaison des métadonnées de tâches Amazon ECS version 4 pour les tâches sur Fargate

Considérations

Tenez compte des éléments suivants lorsque vous utilisez la surveillance d'exécution :

- La surveillance d'exécution a un coût associé. Pour plus d'informations, voir Amazon GuardDuty Tarification.
- La surveillance d'exécution n'est pas prise en charge sur Amazon ECS Anywhere.
- La surveillance d'exécution n'est pas prise en charge pour le système d'exploitation Windows.
- Lorsque vous utilisez Amazon ECS Exec sur Fargate, vous devez spécifier le nom du conteneur car GuardDuty L'agent de sécurité fonctionne comme un conteneur sidecar.
- Vous ne pouvez pas utiliser Amazon ECS Exec sur GuardDuty conteneur sidecar pour agent de sécurité.
- L'utilisateur IAM qui contrôle la surveillance d'exécution au niveau du cluster doit disposer des autorisations IAM appropriées pour l'étiquetage. Pour plus d'informations, voir Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises dans le Guide de l'utilisateur IAM.
- Les tâches Fargate doivent utiliser un rôle d'exécution de tâche. Ce rôle accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer les GuardDuty agent de sécurité, qui est stocké dans un référentiel privé Amazon ECR, en votre nom.
- La surveillance d'exécution n'est pas prise en charge pour les applications exécutées sur les instances gérées Amazon ECS.

Utilisation des ressources

La balise que vous ajoutez au cluster est prise en compte dans le quota de balises du cluster.

Le GuardDuty le conteneur agent sidecar n'est pas pris en compte dans le quota de définition des conteneurs par tâche.

Comme pour la plupart des logiciels de sécurité, il y a une légère surcharge pour GuardDuty. Pour plus d'informations sur les limites de mémoire de Fargate, voir Limites du processeur et de la mémoire dans le GuardDuty Guide de l'utilisateur. Pour plus d'informations sur les limites de mémoire d'Amazon EC2, consultez Limite de processeur et de mémoire pour GuardDuty agent.

Surveillance d’exécution pour les charges de travail Amazon ECS Fargate

Si vous utilisez des instances de conteneur EC2, vous devez configurer manuellement la surveillance d’exécution. Pour plus d’informations, consultez Surveillance de l’exécution des charges de travail EC2 sur Amazon ECS.

Tu peux avoir GuardDuty gérez l'agent de sécurité sur vos instances de conteneur. Cette option n’est disponible que pour Fargate. Cette option (GuardDuty gestion des agents) est disponible dans GuardDuty

Quand vous utilisez GuardDuty gestion des agents, GuardDuty effectue les opérations suivantes :

- Crée des points de terminaison VPC pour GuardDuty pour chaque VPC hébergeant un cluster.
- Récupère et installe la dernière version GuardDuty agent de sécurité en tant que conteneur annexe pour toutes les nouvelles tâches autonomes de Fargate et les nouveaux déploiements de services.

  Un nouveau déploiement de service a lieu la première fois que vous lancez un service ou lorsque vous mettez à jour un service existant à l’aide de l’option Forcer un nouveau déploiement.

Activation de la surveillance d’exécution pour Amazon ECS

Vous pouvez configurer GuardDuty pour gérer automatiquement l'agent de sécurité pour tous vos clusters Fargate.

Conditions préalables

Les conditions préalables suivantes doivent être remplies pour pouvoir utiliser la surveillance d’exécution :

- La version de plateforme Fargate doit être `1.4.0` ou ultérieure pour Linux.
- Rôles et autorisations IAM pour Amazon ECS :
  - Les tâches Fargate doivent utiliser un rôle d’exécution de tâche. Ce rôle accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer les GuardDuty agent de sécurité en votre nom. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

  - Vous contrôlez la surveillance d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, voir Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises dans le Guide de l'utilisateur IAM.
- Connexion au référentiel Amazon ECR :

  Le GuardDuty l'agent de sécurité est stocké dans un référentiel Amazon ECR. Chaque tâche autonome ou tâche de service doit avoir un accès au référentiel. Vous pouvez utiliser l'une des options suivantes :

  - Pour les tâches dans les sous-réseaux publics, vous pouvez soit utiliser une adresse IP publique pour la tâche, soit créer un point de terminaison de VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
  - Pour les tâches dans des sous-réseaux privés, vous pouvez utiliser une passerelle de traduction d'adresses réseau (NAT) ou créer un point de terminaison de VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute.

    Pour plus d'informations, consultez la section Sous-réseau privé et passerelle NAT.
- Vous devez avoir `AWSServiceRoleForAmazonGuardDuty` rôle pour GuardDuty. Pour plus d'informations, voir Service-linked autorisations de rôle pour GuardDuty dans le Amazon GuardDuty Guide de l'utilisateur.
- Tous les fichiers que vous souhaitez protéger avec la surveillance d'exécution doivent être accessibles par l'utilisateur racine. Si vous avez modifié manuellement les autorisations d'un fichier, vous devez le définir sur `755`.

Les conditions préalables suivantes doivent être remplies pour utiliser la surveillance d'exécution sur les instances de conteneur EC2 :

- Vous devez utiliser la version `20230929` ou version ultérieure d'Amazon ECS-AMI.
- Vous devez exécuter l'agent Amazon ECS version `1.77` ou ultérieure sur les instances de conteneur.
- Vous devez utiliser la version du noyau `5.10` ou une version ultérieure.

- Pour plus d'informations sur les systèmes d'exploitation et les architectures Linux pris en charge, voir Quels sont les modèles d'exploitation et les charges de travail GuardDuty Support de surveillance de l'exécution.
- Vous pouvez utiliser Systems Manager pour gérer vos instances de conteneur. Pour plus d'informations, consultez la section Configuration de Systems Manager pour des instances EC2 dans le Guide de l'utilisateur AWS Systems Manager Session Manager .

Procédure

Vous activez la surveillance du temps d'exécution dans GuardDuty. Pour plus d'informations sur l'activation de cette fonctionnalité, voir Activer la surveillance du temps d'exécution dans le Amazon GuardDuty Guide de l'utilisateur.

Ajout de la surveillance d'exécution aux tâches Amazon ECS Fargate existantes

Lorsque vous activez la surveillance d'exécution, toutes les nouvelles tâches autonomes et les nouveaux déploiements de service dans le cluster sont automatiquement protégés. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées.

Conditions préalables

1. Activez la surveillance d'exécution. Pour plus d'informations, consultez Activation de la surveillance d'exécution pour Amazon ECS.
2. Les tâches Fargate doivent utiliser un rôle d'exécution de tâche. Ce rôle accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer les GuardDuty agent de sécurité en votre nom. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Procédure

- Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :
  - Pour les tâches autonomes, arrêtez les tâches, puis redémarrez-les. Pour plus d'informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.
  - Pour les tâches qui font partie d'un service, mettez à jour le service avec l'option « Forcer un nouveau déploiement ». Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Suppression de la surveillance d'exécution d'un cluster Amazon ECS

Vous souhaiterez peut-être exclure certains clusters de la protection, par exemple les clusters que vous utilisez pour les tests. Cela provoque GuardDuty pour effectuer les opérations suivantes sur les ressources du cluster :

- Ne déployez plus le GuardDuty agent de sécurité pour les nouvelles tâches autonomes de Fargate ou les déploiements de nouveaux services.

  Afin de préserver la contrainte d'immuabilité, les tâches et les déploiements existants avec la surveillance d'exécution activée ne sont pas affectés.
- Arrêt de la facturation et refus des événements d'exécution pour les tâches.

Procédure

Effectuez les étapes suivantes pour supprimer la surveillance d'exécution d'un cluster.

1. Utilisez la console Amazon ECS ou AWS CLI pour régler le `GuardDutyManaged` touche de balise sur le cluster pour `false`. Pour plus d'informations, consultez les sections Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API. Utilisez les valeurs suivantes pour les balises.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé = `GuardDutyManaged`, Valeur = `false`
2. Supprimez le GuardDuty Point de terminaison VPC pour le cluster. Pour plus d'informations sur la suppression d'un point de terminaison d'interface, consultez la section Suppression d'un point de terminaison d'interface dans le Guide de l'utilisateur AWS PrivateLink .

Suppression de la surveillance d'exécution pour Amazon ECS d'un compte

Lorsque vous ne souhaitez plus utiliser Runtime Monitoring, désactivez la fonctionnalité dans GuardDuty. Pour plus d'informations sur la désactivation de cette fonctionnalité, voir Activer la surveillance du temps d'exécution dans le Amazon GuardDuty Guide de l'utilisateur.

GuardDuty effectue les opérations suivantes :

- Supprime les points de terminaison VPC pour GuardDuty pour chaque VPC hébergeant un cluster.
- Ne déploie plus le GuardDuty agent de sécurité pour les nouvelles tâches autonomes de Fargate ou les déploiements de nouveaux services.

  Afin de préserver la contrainte d'immuabilité, les tâches et les déploiements existants ne sont pas affectés tant qu'ils ne sont pas arrêtés, répliqués ou mis à l'échelle.
- Arrêt de la facturation et refus des événements d'exécution pour les tâches.

Surveillance de l'exécution des charges de travail EC2 sur Amazon ECS

Utilisez cette option lorsque vous utilisez des instances EC2 pour votre capacité ou lorsque vous avez besoin d'un contrôle granulaire de la surveillance d'exécution au niveau du cluster sur Fargate.

Vous allouez les clusters pour la surveillance d'exécution en ajoutant une balise prédéfinie.

Pour les instances de conteneur EC2, vous téléchargez, installez et gérez le GuardDuty agent de sécurité.

Pour Fargate, GuardDuty gère l'agent de sécurité en votre nom.

Activation de la surveillance d'exécution pour Amazon ECS

Vous pouvez activer la surveillance d'exécution pour les clusters dotés d'instances EC2 ou lorsque vous avez besoin d'un contrôle granulaire de la surveillance d'exécution au niveau du cluster sur Fargate.

Les conditions préalables suivantes doivent être remplies pour pouvoir utiliser la surveillance d'exécution :

- La version de plateforme Fargate doit être `1.4.0` ou ultérieure pour Linux.
- Rôles et autorisations IAM pour Amazon ECS :
  - Les tâches Fargate doivent utiliser un rôle d'exécution de tâche. Ce rôle accorde aux tâches l'autorisation de récupérer, de mettre à jour et de gérer les GuardDuty agent de sécurité en votre nom. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.
  - Vous contrôlez la surveillance d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations

explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, voir Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises dans le Guide de l'utilisateur IAM.

- Connexion au référentiel Amazon ECR :

  Le GuardDuty l'agent de sécurité est stocké dans un référentiel Amazon ECR. Chaque tâche autonome ou tâche de service doit avoir un accès au référentiel. Vous pouvez utiliser l'une des options suivantes :

  - Pour les tâches dans les sous-réseaux publics, vous pouvez soit utiliser une adresse IP publique pour la tâche, soit créer un point de terminaison de VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute. Pour plus d'informations, consultez Points de terminaison d'un VPC d'interface Amazon ECR (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.
  - Pour les tâches dans des sous-réseaux privés, vous pouvez utiliser une passerelle de traduction d'adresses réseau (NAT) ou créer un point de terminaison de VPC pour Amazon ECR dans le sous-réseau sur lequel la tâche s'exécute.

    Pour plus d'informations, consultez la section Sous-réseau privé et passerelle NAT.

- Vous devez avoir `AWSServiceRoleForAmazonGuardDuty` rôle pour GuardDuty. Pour plus d'informations, voir Service-linked autorisations de rôle pour GuardDuty dans le Amazon GuardDuty Guide de l'utilisateur.
- Tous les fichiers que vous souhaitez protéger avec la surveillance d'exécution doivent être accessibles par l'utilisateur racine. Si vous avez modifié manuellement les autorisations d'un fichier, vous devez le définir sur `755`.

Les conditions préalables suivantes doivent être remplies pour utiliser la surveillance d'exécution sur les instances de conteneur EC2 :

- Vous devez utiliser la version `20230929` ou version ultérieure d'Amazon ECS-AMI.
- Vous devez exécuter l'agent Amazon ECS version `1.77` ou ultérieure sur les instances de conteneur.
- Vous devez utiliser la version du noyau `5.10` ou une version ultérieure.
- Pour plus d'informations sur les systèmes d'exploitation et les architectures Linux pris en charge, voir Quels sont les modèles d'exploitation et les charges de travail GuardDuty Support de surveillance de l'exécution.

- Vous pouvez utiliser Systems Manager pour gérer vos instances de conteneur. Pour plus d'informations, consultez la section Configuration de Systems Manager pour des instances EC2 dans le Guide de l'utilisateur AWS Systems Manager Session Manager .

Vous activez la surveillance du temps d'exécution dans GuardDuty. Pour plus d'informations sur l'activation de cette fonctionnalité, voir Activer la surveillance du temps d'exécution dans le Amazon GuardDuty Guide de l'utilisateur.

Ajout de la surveillance d'exécution à un cluster Amazon ECS

Configurez la surveillance de l'exécution pour le cluster, puis installez GuardDuty agent de sécurité sur vos instances de conteneur EC2.

Conditions préalables

1. Activez la surveillance d'exécution. Pour plus d'informations, consultez Activation de la surveillance d'exécution pour Amazon ECS.
2. Vous contrôlez la surveillance d'exécution pour un cluster à l'aide d'une balise prédéfinie. Si vos politiques d'accès limitent l'accès en fonction de balises, vous devez accorder des autorisations explicites à vos utilisateurs IAM pour étiqueter les clusters. Pour plus d'informations, voir Tutoriel IAM : définir les autorisations d'accès AWS ressources basées sur des balises dans le Guide de l'utilisateur IAM.

Procédure

Effectuez les opérations suivantes pour ajouter la surveillance d'exécution à un cluster.

1. Créez un point de terminaison VPC pour GuardDuty pour chaque VPC de cluster. Pour plus d'informations, voir Création manuelle d'un point de terminaison Amazon VPC dans le GuardDuty Guide de l'utilisateur.
2. Configurez les instances de conteneur EC2.

    a. Mettez à jour l'agent Amazon ECS vers la version `1.77` ou une version ultérieure sur les instances de conteneur EC2 du cluster. Pour plus d'informations, consultez Mise à jour de l'agent de conteneur Amazon ECS.

    b. Installez le GuardDuty agent de sécurité sur les instances de conteneur EC2 du cluster. Pour plus d'informations, voir Gestion manuelle de l'agent de sécurité sur une instance Amazon EC2 dans le GuardDuty Guide de l'utilisateur.

Toutes les tâches et tous les déploiements nouveaux et existants sont immédiatement protégés car GuardDuty l'agent de sécurité s'exécute en tant que processus sur l'instance de conteneur EC2.

3. Utilisez la console Amazon ECS ou AWS CLI pour régler le `GuardDutyManaged` touche de balise sur le cluster pour `true`. Pour plus d'informations, consultez les sections Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API. Utilisez les valeurs suivantes pour les balises.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé = `GuardDutyManaged`, Valeur = `true`

Ajout de la surveillance d'exécution aux tâches Amazon ECS existantes

Lorsque vous activez la surveillance d'exécution, toutes les nouvelles tâches autonomes et les nouveaux déploiements de service dans le cluster sont automatiquement protégés. Afin de préserver la contrainte d'immuabilité, les tâches existantes ne sont pas affectées.

Conditions préalables

- Activez la surveillance d'exécution. Pour plus d'informations, consultez Activation de la surveillance d'exécution pour Amazon ECS.

Procédure

- Pour protéger immédiatement une tâche, vous devez effectuer l'une des actions suivantes :
  - Pour les tâches autonomes, arrêtez les tâches, puis redémarrez-les. Pour plus d'informations, consultez Arrêt d'une tâche Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.
  - Pour les tâches qui font partie d'un service, mettez à jour le service avec l'option « Forcer un nouveau déploiement ». Pour plus d'informations, consultez Mettre à jour un service Amazon ECS.

Suppression de la surveillance d'exécution d'un cluster Amazon ECS

Vous pouvez supprimer la surveillance d'exécution d'un cluster. Cela provoque GuardDuty pour arrêter de surveiller toutes les ressources du cluster.

Pour supprimer la surveillance d'exécution d'un cluster

1. Utilisez la console Amazon ECS ou AWS CLI pour régler le `GuardDutyManaged` touche de balise sur le cluster pour `false`. Pour plus d'informations, consultez les sections Mise à jour d'un cluster ou Utilisation des balises à l'aide de la CLI ou de l'API.

   > **Note**
   >
   > La clé et la valeur distinguent les majuscules et minuscules et doivent correspondre exactement aux chaînes.

   Clé = `GuardDutyManaged`, Valeur = `false`
2. Désinstallez le GuardDuty agent de sécurité sur vos instances de conteneur EC2 dans le cluster.

   Pour plus d'informations, voir Désinstallation manuelle de l'agent de sécurité dans le GuardDuty Guide de l'utilisateur.
3. Supprimez le GuardDuty Point de terminaison VPC pour chaque VPC de cluster. Pour plus d'informations sur la suppression d'un point de terminaison d'interface, consultez la section Suppression d'un point de terminaison d'interface dans le Guide de l'utilisateur AWS PrivateLink .

Mettre à jour le GuardDuty agent de sécurité sur vos instances de conteneur Amazon ECS

Pour plus d'informations sur la mise à jour du GuardDuty agent de sécurité sur vos instances de conteneur EC2, voir Mise à jour GuardDuty agent de sécurité dans le Amazon GuardDuty Guide de l'utilisateur.

Suppression de la surveillance d'exécution pour Amazon ECS d'un compte

Lorsque vous ne souhaitez plus utiliser Runtime Monitoring, désactivez la fonctionnalité dans GuardDuty. Pour plus d'informations sur la désactivation de cette fonctionnalité, voir Activer la surveillance du temps d'exécution dans le Amazon GuardDuty Guide de l'utilisateur.

Supprimez la surveillance d'exécution de tous les clusters. Pour plus d'informations, consultez Suppression de la surveillance d'exécution d'un cluster Amazon ECS.

Résolution des problèmes de surveillance d'exécution

Vous devrez peut-être résoudre les problèmes ou vérifier que la surveillance d'exécution est activée et exécutée sur vos tâches et vos conteneurs.

Rubriques

- Comment savoir si la surveillance d'exécution est active sur mon compte ?
- Comment savoir si la surveillance d'exécution est active sur un cluster ?
- Comment puis-je savoir si GuardDuty Un agent de sécurité exécute une tâche Fargate ?
- Comment puis-je savoir si GuardDuty l'agent de sécurité s'exécute sur une instance de conteneur EC2 ?
- Que se passe-t-il lorsqu'il n'existe aucun rôle d'exécution de tâche pour une tâche exécutée sur le cluster ?
- Comment puis-je savoir si je dispose des autorisations appropriées pour étiqueter des clusters à des fins de surveillance d'exécution ?
- Que se passe-t-il en cas d'absence de connexion Amazon ECR ?
- Comment puis-je corriger les erreurs liées au manque de mémoire sur mes tâches Fargate après avoir activé la surveillance d'exécution ?

Comment savoir si la surveillance d'exécution est active sur mon compte ?

Dans la console Amazon ECS, les informations se trouvent sur la page Paramètres du compte.

Vous pouvez également exécuter `list-account-settings` avec l'option `effective-settings`.

```
aws ecs list-account-settings --effective-settings
```

Sortie

Le paramètre dont le nom est défini sur `guardDutyActivate` et la valeur sur on indique que le compte est configuré. Vous devez vérifier auprès de votre GuardDuty administrateur pour voir si la gestion est automatique ou manuelle.

```
{
    "setting": {
        "name": "guardDutyActivate",
        "value": "enabled",
        "principalArn": "arn:aws:iam::123456789012:root",
        "type": "aws-managed"
    }
}
```

Comment savoir si la surveillance d'exécution est active sur un cluster ?

Vous pouvez consulter les statistiques de couverture dans GuardDuty console. Cela inclut les informations relatives aux ressources Amazon ECS associées à votre propre compte ou à vos comptes membres, soit le pourcentage de clusters sains par rapport à l'ensemble des clusters dans l' Région AWS sélectionnée. Cela inclut la couverture des clusters utilisant Fargate et EC2. Pour plus d'informations, voir Révision des statistiques de couverture dans le Amazon GuardDuty Guide de l'utilisateur.

Comment puis-je savoir si GuardDuty Un agent de sécurité exécute une tâche Fargate ?

Le GuardDuty L'agent de sécurité fonctionne comme un conteneur sidecar pour les tâches Fargate.

Dans la console Amazon ECS, le sidecar s'affiche sous Conteneurs sur la page Détails de la tâche.

Vous pouvez exécuter `describe-tasks` et rechercher le conteneur dont le nom est défini sur `aws-gd-agent` et le LastStatus défini sur `RUNNING`.

Voici un exemple de la sortie du cluster par défaut pour la tâche `aws:ecs:us-east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE`.

```
aws ecs describe-tasks --cluster default --tasks aws:ecs:us-
east-1:123456789012:task/0b69d5c0-d655-4695-98cd-5d2d5EXAMPLE
```

Sortie

Le conteneur nommé `gd-agent` est dans l'état `RUNNING`.

```
"containers": [
      {
        "containerArn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-
f057-467b-a079-96167EXAMPLE",
        "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/0b69d5c0-
d655-4695-98cd-5d2d5EXAMPLE",
        "lastStatus": "RUNNING",
        "healthStatus": "UNKNOWN",
        "memory": "string",
        "name": "aws-gd-agent"
      }
    ]
```

Comment puis-je savoir si GuardDuty l'agent de sécurité s'exécute sur une instance de conteneur EC2 ?

Exécutez la commande suivante pour afficher l'état :

```
sudo systemctl status amazon-guardduty-agent
```

Le fichier journal se trouve à l'emplacement suivant :

```
/var/log/amzn-guardduty-agent
```

Que se passe-t-il lorsqu'il n'existe aucun rôle d'exécution de tâche pour une tâche exécutée sur le cluster ?

Pour les tâches Fargate, la tâche commence sans GuardDuty conteneur sidecar pour agent de sécurité. Le GuardDuty le tableau de bord indiquera que la tâche n'est pas protégée dans le tableau de bord des statistiques de couverture.

Comment puis-je savoir si je dispose des autorisations appropriées pour étiqueter des clusters à des fins de surveillance d'exécution ?

Pour étiqueter un cluster, vous devez avoir l'action `ecs:TagResource` à la fois pour `CreateCluster` et `UpdateCluster`.

L'extrait suivant représente un exemple de politique.

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
             "ecs:CreateAction" : [ "CreateCluster", "UpdateCluster" ]
          }
       }
    }
  ]
}
```

Que se passe-t-il en cas d’absence de connexion Amazon ECR ?

Pour les tâches Fargate, la tâche commence sans GuardDuty conteneur sidecar pour agent de sécurité. Le GuardDuty le tableau de bord indiquera que la tâche n'est pas protégée dans le tableau de bord des statistiques de couverture.

Comment puis-je corriger les erreurs liées au manque de mémoire sur mes tâches Fargate après avoir activé la surveillance d’exécution ?

Le GuardDuty l'agent de sécurité est un processus léger. Cependant, le processus consomme toujours des ressources en fonction de l’importance de la charge de travail. Nous vous recommandons d'utiliser des outils de suivi des ressources de conteneurs, tels que Amazon CloudWatch Container Insights sur scène GuardDuty déploiements dans votre cluster. Ces outils vous aident à découvrir le profil de consommation du GuardDuty agent de sécurité pour vos applications. Vous pouvez ensuite ajuster la taille de votre tâche Fargate, si nécessaire, pour éviter tout risque de manque de mémoire.

Surveillance des conteneurs Amazon ECS avec ECS Exec

Avec Amazon ECS Exec, vous pouvez interagir directement avec les conteneurs sans avoir besoin d'interagir avec le système d'exploitation du conteneur hôte, d'ouvrir des ports entrants ou de gérer les clés SSH au préalable. Vous pouvez utiliser ECS Exec pour exécuter des commandes dans

ou obtenir un shell vers un conteneur s'exécutant sur une instance Amazon EC2 ou sur AWS Fargate. Cela facilite la collecte des informations de diagnostic et la résolution rapide des erreurs. Par exemple, dans le cadre d'un développement, vous pouvez utiliser ECS Exec pour interagir facilement avec différents processus dans vos conteneurs et dépanner vos applications. Et dans les scénarios de production, vous pouvez l'utiliser pour obtenir un accès d'urgence à vos conteneurs afin de déboguer les problèmes.

Vous pouvez exécuter des commandes dans un conteneur Linux ou Windows en cours d'exécution à l'aide d'ECS Exec depuis l'API Amazon ECS, AWS Command Line Interface (AWS CLI), AWS SDK, ou AWS Interface de ligne de commande Copilote. Pour plus de détails sur l'utilisation d'ECS Exec, ainsi qu'une présentation vidéo, utilisez AWS Interface de ligne de commande Copilot, consultez le Copilote GitHub documentation.

Vous pouvez également utiliser ECS Exec pour appliquer des politiques de contrôle d'accès plus strictes. En activant cette fonction de manière sélective, vous pouvez contrôler qui peut exécuter des commandes et sur quelles tâches. Avec un journal de chaque commande et de leur sortie, vous pouvez utiliser ECS Exec pour voir quelles tâches ont été exécutées et vous pouvez utiliser CloudTrail pour vérifier qui a accédé à un conteneur.

Considérations

Tenez compte des éléments suivants lorsque vous utilisez ECS Exec :

- ECS Exec risque de ne pas fonctionner comme prévu s'il est exécuté sur des systèmes d'exploitation non pris en charge par Systems Manager. Pour plus d'informations sur les systèmes d'exploitation pris en charge, consultez la section Types de systèmes d'exploitation dans le Guide de l'utilisateur AWS Systems Manager .
- ECS Exec est pris en charge pour les tâches exécutées sur l'infrastructure suivante :
  - Conteneurs Linux sur Amazon EC2 sur n'importe quel Amazon ECS-optimized AMI, y compris Bottlerocket
  - Conteneurs Linux et Windows sur des instances externes (Amazon ECS Anywhere)
  - Conteneurs Linux et Windows sur AWS Fargate
  - Conteneurs Windows sur Amazon EC2 sur les appareils Windows Amazon suivants ECS-optimized AMI (avec la version Container Agent) `1.56` ou plus tard) :
    - Amazon ECS-optimized AMI complète pour Windows Server 2022
    - Amazon ECS-optimized AMI principale pour Windows Server 2022
    - Amazon ECS-optimized AMI complète pour Windows Server 2019

    - Amazon ECS-optimized AMI principale pour Windows Server 2019
    - Amazon ECS-optimized AMI principale Windows Server 20H2
- Si vous avez configuré un proxy HTTP pour votre tâche, définissez la variable d'environnement `NO_PROXY` sur `"NO_PROXY=169.254.169.254,169.254.170.2"` afin de contourner le proxy pour les métadonnées de l'instance EC2 et le trafic des rôles IAM. Si vous ne configurez pas la variable d'environnement `NO_PROXY`, des échecs peuvent survenir lors de la récupération des métadonnées de l'instance ou des informations d'identification du rôle IAM à partir du point de terminaison des métadonnées du conteneur. La définition de la variable d'environnement `NO_PROXY` comme recommandé filtre les métadonnées et le trafic IAM afin que les requêtes `169.254.169.254 and 169.254.170.2` ne passent pas par le proxy `HTTP`.
- ECS Exec et Amazon VPC
    - Si vous utilisez l'interface des points de terminaison Amazon VPC avec Amazon ECS, vous devez créer l'interface des points de terminaison Amazon VPC pour le Gestionnaire de sessions Systems Manager (`ssmmessages`). Pour plus d'informations sur les points de terminaison VPC de Systems Manager, consultez Utiliser AWS PrivateLink pour configurer un point de terminaison VPC pour Session Manager dans le AWS Systems Manager Guide de l'utilisateur.
    - Si vous utilisez une interface entre les points de terminaison Amazon VPC et Amazon ECS, et que vous utilisez AWS KMS key pour le chiffrement, vous devez créer l'interface Amazon VPC endpoint pour AWS KMS key. Pour plus d'informations, consultez Configuration à AWS KMS key via un point de terminaison de VPC dans le Guide du développeur AWS Key Management Service .
    - Lorsque vous avez des tâches exécutées sur des instances Amazon EC2, utilisez le mode réseau `awsvpc`. Si vous n'avez pas accès à Internet (par exemple, si vous n'êtes pas configuré pour utiliser une passerelle NAT), vous devez créer les points de terminaison Amazon VPC d'interface pour le Gestionnaire de session Systems Manager (`ssmmessages`). Pour plus d'informations sur les considérations relatives au mode réseau `awsvpc`, veuillez consulter Considérations (langue française non garantie). Pour plus d'informations sur les points de terminaison VPC de Systems Manager, consultez Utiliser AWS PrivateLink pour configurer un point de terminaison VPC pour Session Manager dans le AWS Systems Manager Guide de l'utilisateur.
- Amazon ECS Exec n'est pas pris en charge pour les tâches exécutées dans IPv6-only configuration. Pour plus d'informations sur l'exécution de tâches dans un IPv6-only configuration, voir Options de mise en réseau des tâches Amazon ECS pour Fargate et Options de mise en réseau des tâches Amazon ECS pour EC2.
- ECS Exec et SSM

  - Votre tâche nécessite un rôle IAM doté des autorisations requises pour SSM Session Manager. Pour les tâches sur Amazon EC2, si aucun rôle de tâche n'est configuré, le rôle d'instance de l'instance Amazon EC2 sous-jacente est utilisé à la place. Pour plus d'informations, consultez Autorisations ECS Exec.
  - Lorsqu'un utilisateur exécute des commandes sur un conteneur à l'aide d'ECS Exec, ces commandes sont exécutées en tant qu'utilisateur `root`. Le SSM Agent et ses processus enfants s'exécutent en tant qu'utilisateur racine même lorsque vous spécifiez un ID d'utilisateur pour le conteneur.
  - L'agent SSM exige qu'il soit possible d'écrire dans le système de fichiers du conteneur pour créer les répertoires et fichiers requis. Par conséquent, il n'est pas possible de rendre le système de fichiers racine en lecture seule à l'aide du paramètre de définition de tâche `readonlyRootFilesystem` ou de toute autre méthode.
  - Il est possible de démarrer des sessions SSM en dehors de l'action `execute-command`, mais les sessions ne sont pas journalisées et sont comptabilisées afin de vérifier qu'elles ne dépassent pas la limite de session. Nous vous recommandons de limiter cet accès en refusant l'action `ssm:start-session` à l'aide d'une stratégie IAM. Pour plus d'informations, consultez Limitation de l'accès à l'action StartSession.
- Les fonctionnalités suivantes s'exécutent en tant que conteneur sidecar. Par conséquent, vous devez spécifier le nom du conteneur sur lequel exécuter la commande.
  - Surveillance d'exécution
  - Connexion au service
- Les utilisateurs peuvent exécuter toutes les commandes qui sont disponibles dans le contexte de conteneur. Les actions suivantes peuvent entraîner des processus orphelins et zombies : arrêt du processus principal du conteneur, arrêt de l'agent de commande et suppression des dépendances. Pour nettoyer les processus zombie, nous vous recommandons d'ajouter l'indicateur `initProcessEnabled` à votre définition de tâche.
- ECS Exec utilise une partie de l'UC et de la mémoire. Vous devez en tenir compte lorsque vous spécifiez les allocations de ressources CPU et mémoire dans votre définition de tâche.
- Vous devez utiliser AWS CLI version `1.22.3` ou plus tard ou AWS CLI version `2.3.6` ou plus tard. Pour plus d'informations sur la mise à jour du AWS CLI, voir Installation ou mise à jour de la dernière version du AWS CLI dans le AWS Command Line Interface Guide de l'utilisateur version 2.

- Vous ne pouvez avoir qu'une seule session ECS Exec par espace de noms d'ID de processus (PID). Si vous partagez un espace de noms PID dans une tâche, vous ne pouvez démarrer des sessions ECS Exec que dans un seul conteneur.
- La session ECS Exec a une valeur de délai d'inactivité de 20 minutes. Cette valeur ne peut pas être modifiée.
- Vous ne pouvez pas activer ECS Exec pour les tâches existantes. Vous ne pouvez l'activer que pour les nouvelles tâches.
- Vous ne pouvez pas utiliser ECS Exec lorsque vous utilisez `run-task` pour lancer une tâche sur un cluster qui utilise la mise à l'échelle gérée avec placement asynchrone (lancement d'une tâche sans instance).
- Vous ne pouvez pas exécuter ECS Exec sur des conteneurs Microsoft Nano Server.

Architecture

ECS Exec utilise AWS Systems Manager (SSM) Session Manager pour établir une connexion avec le conteneur en cours d'exécution et utilise Gestion des identités et des accès AWS politiques (IAM) pour contrôler l'accès aux commandes en cours d'exécution dans un conteneur en cours d'exécution. Ceci est rendu possible par le montage lié des fichiers binaires du SSM Agent nécessaires dans le conteneur. L'Amazon ECS ou AWS Fargate l'agent est chargé de démarrer l'agent principal SSM à l'intérieur du conteneur à côté du code de votre application. Pour plus d'informations, consultez Systems Manager Session Manager.

Vous pouvez vérifier quel utilisateur a accédé au conteneur à l'aide du `ExecuteCommand` événement en AWS CloudTrail et enregistrez chaque commande (et son résultat) sur Amazon S3 ou Amazon CloudWatch Journaux. Pour chiffrer les données entre le client local et le conteneur à l'aide de votre propre clé de chiffrement, vous devez fournir le AWS Key Management Service (AWS KMS) clé.

Configuration d'ECS Exec

Pour utiliser ECS Exec, vous devez d'abord activer la fonctionnalité pour vos tâches et services. Vous pouvez ensuite exécuter des commandes dans vos conteneurs.

Modifications facultatives de la définition de tâche

Si vous définissez le paramètre de définition de tâche `initProcessEnabled` sur `true`, le processus init démarre à l'intérieur du conteneur. Cela supprime tous les processus enfants de l'agent SSM zombie trouvés. Voici un exemple.

```
{
    "taskRoleArn": "ecsTaskRole",
    "networkMode": "awsvpc",
    "requiresCompatibilities": [
        "EC2",
        "FARGATE"
    ],
    "executionRoleArn": "ecsTaskExecutionRole",
    "memory": ".5 gb",
    "cpu": ".25 vcpu",
    "containerDefinitions": [
        {
            "name": "amazon-linux",
            "image": "amazonlinux:latest",
            "essential": true,
            "command": ["sleep","3600"],
            "linuxParameters": {
                "initProcessEnabled": true
            }
        }
    ],
    "family": "ecs-exec-task"
}
```

Activation d'ECS Exec pour vos tâches et services

Vous pouvez activer la fonction ECS Exec pour vos services et vos tâches autonomes en spécifiant `--enable-execute-command` drapeau lors de l'utilisation de l'un des indicateurs suivants AWS CLI commandes : `create-service`, `update-service`, `start-task`, ou `run-task`.

Par exemple, si vous exécutez la commande suivante, la fonctionnalité ECS Exec est activée pour un service nouvellement créé. Pour plus d'informations sur la création de services, consultez create-service.

```
aws ecs create-service \
    --cluster cluster-name \
    --task-definition task-definition-name \
    --enable-execute-command \
    --service-name service-name \
    --launch-type FARGATE \
```

```
    --network-configuration
"awsvpcConfiguration={subnets=[subnet-12344321],securityGroups=[sg-12344321],assignPublicIp=EN
\
   --desired-count 1
```

Après avoir activé ECS Exec pour une tâche, vous pouvez exécuter la commande suivante pour confirmer que la tâche est prête à être utilisée. Si la propriété `lastStatus` de `ExecuteCommandAgent` est répertoriée comme `RUNNING` et que la propriété `enableExecuteCommand` est définie sur `true`, alors votre tâche est prête.

```
aws ecs describe-tasks \
    --cluster cluster-name \
    --tasks task-id
```

L'extrait de sortie suivant est un exemple de ce que vous pouvez voir.

```
{
    "tasks": [
        {
            ...
            "containers": [
                {
                    ...
                    "managedAgents": [
                        {
                            "lastStartedAt": "2021-03-01T14:49:44.574000-06:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                    ]
                }
            ],
            ...
            "enableExecuteCommand": true,
            ...
        }
    ]
}
```

Exécution de commandes avec ECS Exec

Journalisation à l'aide d'ECS Exec

Vous pouvez configurer la journalisation des sessions ECS Exec afin de capturer les commandes et leur sortie à des fins d'audit et de résolution des problèmes.

Activation de la journalisation et de l'audit dans vos tâches et services

> **Important**
>
> Pour plus d'informations sur CloudWatch tarification, voir CloudWatch Tarification.
> Amazon ECS propose également des métriques de surveillance sans coûts supplémentaires. Pour plus d'informations, consultez Surveillez Amazon ECS à l'aide CloudWatch.

Amazon ECS fournit une configuration par défaut pour les commandes de journalisation exécutées à l'aide d'ECS Exec. Par défaut, les journaux sont envoyés à CloudWatch Journaux à l'aide du `awslogs` pilote de journal configuré dans votre définition de tâche. Si vous souhaitez fournir une configuration personnalisée, AWS CLI prend en charge un `--configuration` drapeau pour les deux `create-cluster` et `update-cluster` commandes. L'image du conteneur nécessite `script` et `cat` à installer afin que les journaux de commandes soient chargés correctement sur Amazon S3 ou CloudWatch Journaux. Pour de plus amples informations sur la création de clusters, consultez create-cluster.

> **Note**
>
> Cette configuration ne gère que la journalisation de la session `execute-command`. Cela n'affecte pas la journalisation de votre application.

L'exemple suivant crée un cluster, puis enregistre la sortie dans votre CloudWatch Journaux LogGroup nommé `cloudwatch-log-group-name` et votre compartiment Amazon S3 nommé `s3-bucket-name`.

Vous devez utiliser un AWS KMS clé gérée par le client pour chiffrer le groupe de journaux lorsque vous définissez `CloudWatchEncryptionEnabled` option pour `true`. Pour plus d'informations sur le chiffrement du groupe de journaux, consultez Chiffrez les données du journal dans CloudWatch

Journaux utilisant AWS Key Management Service, dans le Amazon CloudWatch Logs Guide de l'utilisateur.

```
aws ecs create-cluster \
    --cluster-name cluster-name \
    --configuration executeCommandConfiguration="{ \
        kmsKeyId=string, \
        logging=OVERRIDE, \
        logConfiguration={ \
            cloudWatchLogGroupName=cloudwatch-log-group-name, \
            cloudWatchEncryptionEnabled=true, \
            s3BucketName=s3-bucket-name, \
            s3EncryptionEnabled=true, \
            s3KeyPrefix=demo \
        } \
    }"
```

La propriété `logging` détermine le comportement de la capacité de journalisation d'ECS Exec :

- `NONE` : la journalisation est désactivée.
- `DEFAULT` : les journaux sont envoyés au pilote `awslogs` configuré. Si le pilote n'est pas configuré, aucun journal n'est enregistré.
- `OVERRIDE`: les journaux sont envoyés à l'Amazon fourni CloudWatch Journaux LogGroup, un compartiment Amazon S3, ou les deux.

Autorisations IAM requises pour Amazon CloudWatch Journaux pour Amazon S3 Logging

Pour activer la journalisation, le rôle de tâche Amazon ECS référencé dans votre définition de tâche doit disposer d'autorisations supplémentaires. Ces autorisations supplémentaires peuvent être ajoutées en tant que stratégie au rôle de tâche. Ils sont différents selon que vous dirigez ou non vos journaux vers Amazon CloudWatch Logs pour Amazon S3.

Amazon CloudWatch Logs

L'exemple de politique suivant ajoute l'Amazon requis CloudWatch Enregistre les autorisations.

```
{
```

```
   "Version":"2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:DescribeLogGroups"
            ],
            "Resource": "*"
        },
        {
            "Effect": "Allow",
            "Action": [
               "logs:CreateLogStream",
               "logs:DescribeLogStreams",
               "logs:PutLogEvents"
            ],
            "Resource": "arn:aws:logs:us-east-1:111122223333:log-group:/aws/
ecs/cloudwatch-log-group-name:*"
        }
   ]
}
```

Amazon S3

L'exemple suivant de stratégie ajoute les autorisations Amazon S3 requises.

```
{
   "Version":"2012-10-17",
   "Statement": [
        {
            "Effect": "Allow",
            "Action": [
               "s3:GetBucketLocation"
            ],
            "Resource": "*"
        },
        {
           "Effect": "Allow",
           "Action": [
               "s3:GetEncryptionConfiguration"
           ],
           "Resource": "arn:aws:s3:::s3-bucket-name"
```

```
        },
        {
            "Effect": "Allow",
            "Action": [
                "s3:PutObject"
            ],
            "Resource": "arn:aws:s3:::s3-bucket-name/*"
        }
    ]
   }
```

Autorisations IAM requises pour le chiffrement à l'aide des vôtres AWS KMS key (clé KMS)

Par défaut, les données transférées entre votre client local et le conteneur utilisent le cryptage TLS 1.2 qui AWS fournit. Pour chiffrer davantage les données à l'aide de votre propre clé KMS, vous devez créer une clé KMS et ajouter l'autorisation `kms:Decrypt` à votre rôle IAM de tâche. Votre conteneur utilise cette autorisation pour déchiffrer les données. Pour plus d'informations sur la création d'une clé KMS, consultez Création de clés.

Vous ajoutez la politique en ligne suivante à votre rôle IAM de tâche qui nécessite les autorisations AWS KMS . Pour plus d’informations, consultez Autorisations ECS Exec.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt"
            ],
            "Resource": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

Pour que les données soient chiffrées à l'aide de votre propre clé KMS, l'utilisateur ou le groupe utilisant l'action `execute-command` doit être accordée à l'autorisation `kms:GenerateDataKey`.

L'exemple de stratégie suivant pour votre utilisateur ou groupe contient l'autorisation requise pour utiliser votre propre clé KMS. Vous devez spécifier l'Amazon Resource Name (ARN) de votre clé KMS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:GenerateDataKey"
            ],
            "Resource": "arn:aws:kms:us-
east-2:111122223333:key/1234abcd-12ab-34cd-56ef-1234567890ab"
        }
    ]
}
```

Utilisation de stratégies IAM pour limiter l'accès à ECS Exec

Vous limitez l’accès utilisateur à l’action d’API execute-command en utilisant une ou plusieurs des clés de condition de politique IAM suivantes :

- `aws:ResourceTag/`*`clusterTagKey`*
- `ecs:ResourceTag/`*`clusterTagKey`*
- `aws:ResourceTag/`*`taskTagKey`*
- `ecs:ResourceTag/`*`taskTagKey`*
- `ecs:container-name`
- `ecs:cluster`
- `ecs:task`
- `ecs:enable-execute-command`

Avec l'exemple de stratégie IAM suivant, les utilisateurs peuvent exécuter des commandes dans des conteneurs qui s'exécutent dans des tâches avec une balise qui possède une clé `environment` et une valeur `development`, dans un cluster nommé `cluster-name`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ExecuteCommand",
                "ecs:DescribeTasks"
            ],
            "Resource": [
                   "arn:aws:ecs:us-east-1:111122223333:task/cluster-name/*",
                   "arn:aws:ecs:us-east-1:111122223333:cluster/cluster-name"
            ],
            "Condition": {
                "StringEquals": {
                    "ecs:ResourceTag/environment": "development"
                }
            }
        }
    ]
}
```

Avec l'exemple de politique IAM suivant, les utilisateurs ne peuvent pas utiliser l'API `execute-command` lorsque le nom du conteneur est `production-app`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": [
                "ecs:ExecuteCommand"
```

```
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:container-name": "production-app"
                }
            }
        }
    ]
}
```

Avec la stratégie IAM suivante, les utilisateurs ne peuvent lancer des tâches que lorsque ECS Exec est désactivé.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RunTask",
                "ecs:StartTask",
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:enable-execute-command": "false"
                }
            }
        }
    ]
}
```

> **Note**
>
> Étant donné que l'action d'API `execute-command` contient uniquement des ressources de tâche et de cluster dans une reqûete, seules les balises de cluster et de tâche sont évaluées.

Pour plus d'informations sur les clés de condition de stratégie IAM, consultez Actions, ressources et clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service.

Limitation de l'accès à l'action StartSession

Bien qu'il soit possible de démarrer des sessions SSM sur votre conteneur en dehors d'ECS Exec, cela pourrait entraîner la non-journalisation des sessions. Les sessions démarrées en dehors d'ECS Exec sont également comptabilisées dans le quota de session. Nous vous recommandons de limiter cet accès en refusant l'action `ssm:start-session` directement pour vos tâches Amazon ECS à l'aide d'une stratégie IAM. Vous pouvez refuser l'accès à toutes les tâches Amazon ECS ou à des tâches spécifiques en fonction des balises utilisées.

Voici un exemple de stratégie IAM qui refuse l'accès à l'action `ssm:start-session` pour les tâches dans toutes les régions avec un nom de cluster spécifié. Si vous le souhaitez, vous pouvez inclure un caractère générique avec l'option *`cluster-name`*.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": [
                   "arn:aws:ecs:us-east-1:111122223333:task/cluster-name/*"
            ]
        }
    ]
}
```

Voici un exemple de stratégie IAM qui refuse l'accès à l'action `ssm:start-session` sur les ressources dans toutes les régions étiquetées avec la clé de balise `Task-Tag-Key` et la valeur de baliseExec-Task.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Deny",
            "Action": "ssm:StartSession",
            "Resource": "arn:aws:ecs:*:*:task/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Task-Tag-Key": "Exec-Task"
                }
            }
        }
    ]
}
```

Résolution des problèmes ECS Exec

Pour aller plus loin dans la résolution, consultez la section Résolution des problèmes avec Exec.

Exécution de commandes avec ECS Exec

Vous pouvez utiliser Amazon ECS Exec pour collecter des informations de diagnostic relatives à vos conteneurs et résoudre les erreurs rencontrées tout au long du cycle de vie de vos conteneurs.

Conditions préalables

Avant de commencer à utiliser ECS Exec, assurez-vous que vous avez effectué les actions suivantes :

- Prenez connaissance des considérations. Pour plus d'informations, consultez Considérations
- Configurez ECS Exec pour vos tâches et services. Pour plus d'informations, consultez Configuration d'ECS Exec

- Installez et configurez l' AWS CLI. Pour plus d'informations, consultez la section Démarrage avec l' AWS CLI.
- Installez le plug-in Session Manager pour AWS CLI. Pour de plus amples informations, veuillez consulter Install the Session Manager plugin for the AWS CLI.
- Configurez un rôle de tâche avec les autorisations appropriées. Vous devez utiliser un rôle de tâche doté des autorisations appropriées pour ECS Exec. Pour plus d'informations, veuillez consulter Rôle IAM de tâche (langue française non garantie).
- Vérifiez les exigences de version. ECS Exec a des exigences de version selon que vos tâches sont hébergées sur Amazon EC2 ou AWS Fargate :
  - Si vous utilisez Amazon EC2, vous devez utiliser une AMI optimisée pour Amazon ECS publiée après le 20 janvier 2021 avec une version 1.50.2 ou supérieure de l'agent. Pour plus d'informations, consultez AMI optimisées pour Amazon ECS.
  - Si vous utilisez AWS Fargate, vous devez utiliser la version de la plateforme `1.4.0` ou supérieur (Linux) ou `1.0.0` (Fenêtres). Pour plus d'informations, consultez Versions de plateforme AWS Fargate.

Utilisation de la console pour les tâches de service

Vous pouvez utiliser la console pour exécuter des commandes à l'aide d'ECS Exec.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s'affiche.
4. Sur la page des détails du service, choisissez Tâches. Puis choisissez la tâche.
5. Sous Conteneurs, choisissez le conteneur dans lequel vous souhaitez utiliser ECS Exec.
6. Pour exécuter des commandes, procédez de l'une des manières suivantes :

   - Choisissez Se connecter.

     UNE CloudShell session affiche l'endroit où vous pouvez exécuter vos commandes.
   - Cliquez sur la flèche, puis choisissez Copier la commande AWS CLI .

     Vous pouvez ensuite exécuter les commandes localement.

Résultats attendus

Si la connexion est établie, vous devriez voir une invite shell interactive provenant de votre conteneur. Vous pouvez désormais exécuter des commandes directement dans l'environnement du conteneur. Pour mettre fin à la session, choisissez Arrêter la session.

Utilisation de la console pour des tâches autonomes

Vous pouvez utiliser la console pour exécuter des commandes à l'aide d'ECS Exec.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Tâches, choisissez la tâche.

   La page des détails de la tâche s'affiche.
4. Sous Conteneurs, choisissez le conteneur dans lequel vous souhaitez utiliser ECS Exec.
5. Pour exécuter des commandes, procédez de l'une des manières suivantes :

   - Choisissez Se connecter.

     UNE CloudShell session affiche l'endroit où vous pouvez exécuter vos commandes.
   - Cliquez sur la flèche, puis choisissez Copier la commande AWS CLI .

     Vous pouvez ensuite exécuter les commandes localement.

Résultats attendus

Si la connexion est établie, vous devriez voir une invite shell interactive provenant de votre conteneur. Vous pouvez désormais exécuter des commandes directement dans l'environnement du conteneur. Pour mettre fin à la session, choisissez Arrêter la session.

Utilisation du shell de commande

Vous pouvez utiliser le shell de commande pour exécuter des commandes à l'aide d'ECS Exec.

Une fois que vous avez confirmé que `ExecuteCommandAgent` est en cours d'exécution, vous pouvez ouvrir un shell interactif sur votre conteneur à l'aide de la commande suivante. Si votre tâche contient plusieurs conteneurs, vous devez spécifier le nom du conteneur à l'aide de l'indicateur `--container`. Amazon ECS ne prend en charge que le lancement de sessions interactives. Vous devez donc utiliser l'indicateur `--interactive`.

La commande suivante exécutera une commande interactive `/bin/sh` commande contre un conteneur nommé *`container-name`* pour une tâche dont l'ID est *`task-id`*.

Le *`task-id`* est le nom de ressource Amazon (ARN) de la tâche.

```
aws ecs execute-command --cluster cluster-name \
    --task task-id \
    --container container-name \
    --interactive \
    --command "/bin/sh"
```

Résultats attendus

Si la commande aboutit, vous devriez voir une invite shell interactive provenant de votre conteneur. Vous pouvez désormais exécuter des commandes directement dans l'environnement du conteneur. Pour quitter la session, tapez `exit` ou appuyez sur `Ctrl+D`.

Optimiseur de calcul AWS recommandations pour Amazon ECS

Optimiseur de calcul AWS génère des recommandations pour la taille des tâches et des conteneurs Amazon ECS. Pour plus d'informations, consultez Présentation de Optimiseur de calcul AWS dans le Guide de l'utilisateur Optimiseur de calcul AWS .

Recommandations relatives aux tâches et à la taille des conteneurs pour les services Amazon ECS sur AWS Fargate

Optimiseur de calcul AWS génère des recommandations pour les services Amazon ECS sur AWS Fargate. Optimiseur de calcul AWS recommande la taille du processeur des tâches et de la mémoire des tâches ainsi que les tailles de réservation du processeur du conteneur, de la mémoire du conteneur et de la mémoire du conteneur. Ces recommandations sont affichées sur les pages suivantes de la console Compute Optimizer.

- Recommandations pour les services Amazon ECS sur la page Fargate
- Services Amazon ECS sur la page de détails de Fargate

Pour plus d'informations, consultez Affichage des recommandations relatives aux services Amazon ECS sur Fargate dans le Guide de l'utilisateur Optimiseur de calcul AWS .

Serveur Amazon ECS MCP

Le serveur Amazon ECS MCP est un service entièrement géré qui propose AI-powered des expériences pour le développement et les opérations. Le Model Context Protocol (MCP) fournit une interface standardisée qui enrichit les agents et les applications d'IA grâce à des connaissances contextuelles en temps réel de vos charges de travail Amazon ECS, permettant ainsi des réponses et des AI-powered flux de travail plus précis et tenant compte du contexte pour inspecter, dépanner et optimiser vos charges de travail Amazon ECS.

> **Note**
>
> Le serveur Amazon ECS MCP est en version préliminaire et est sujet à modification.

Présentation de

Le serveur Amazon ECS MCP peut être facilement intégré à n'importe quel assistant de codage IA compatible MCP, comme Kiro, ou à des outils tiers tels que Claude Code, Cursor et Cline, afin d'améliorer votre flux de travail de développement.

Pendant le déploiement, il fournit une visibilité en temps réel sur les déploiements de services et l'état de santé des tâches. Pour le développement et les opérations, le serveur simplifie la gestion d'Amazon ECS en fournissant des flux de travail de haut niveau pour surveiller l'état des conteneurs, analyser les événements de service et étudier les problèmes de déploiement. À des fins de débogage et de dépannage, le serveur accélère la résolution des problèmes grâce à des outils de diagnostic complets qui analysent les échecs de tâches, les journaux de conteneurs et les configurations réseau. Ces fonctionnalités sont accessibles par le biais d'interactions en langage naturel, ce qui vous permet d'effectuer des opérations complexes sur les conteneurs de manière plus intuitive et plus efficace.

Le serveur Amazon ECS MCP fournit plusieurs outils que vous pouvez utiliser pour :

- Surveillez les déploiements et les services

  Vérifiez l'état du déploiement, suivez les événements de service et surveillez l'état des conteneurs en temps réel.
- Résoudre les problèmes liés aux conteneurs

Analysez les échecs de tâches, examinez les erreurs d'extraction d'images et examinez les journaux des conteneurs avec des fenêtres temporelles personnalisables.

- Inspectez les configurations réseau

  Passez en revue les configurations de VPC, de sous-réseau et de groupe de sécurité pour vos services Amazon ECS.

- Résoudre les dépendances liées aux ressources

  Identifiez les obstacles qui empêchent la suppression des définitions de tâches et comprenez les relations entre les ressources.

- Évaluer la posture de sécurité

  Collectez les données de configuration de sécurité des clusters à AI-powered des fins d'analyse par rapport aux AWS meilleures pratiques avec des étapes de correction hiérarchisées.

- Accédez à une documentation à jour

  Consultez AWS la documentation actuelle pour obtenir des conseils précis et opportuns sur les fonctionnalités et les configurations d'Amazon ECS.

Le serveur Amazon ECS MCP entièrement géré est hébergé dans le AWS cloud, ce qui élimine la nécessité d'une installation et d'une maintenance locales. Il fournit des fonctionnalités de niveau professionnel telles que des mises à jour et des correctifs automatiques, une sécurité centralisée grâce à l'intégration IAM, une journalisation complète des audits via CloudTrail, ainsi que l'évolutivité, la fiabilité et le support éprouvés de. AWS

Prise en main

Consultez Démarrez avec le serveur Amazon ECS MCP pour démarrer.

Démarrez avec le serveur Amazon ECS MCP

Ce guide explique les étapes à suivre pour configurer et utiliser le serveur Amazon ECS MCP avec vos assistants de code IA. Vous allez apprendre à configurer votre environnement, à vous connecter au serveur et à commencer à gérer vos clusters Amazon ECS via des interactions en langage naturel.

> **Note**
>
> Le serveur Amazon ECS MCP est en version préliminaire et est sujet à modification.

Conditions préalables

Avant de commencer, assurez-vous de disposer des éléments suivants :

- Création d'un AWS compte avec accès à Amazon ECS
- Installé et configuré le AWS CLI avec des informations d'identification
- Python 3.10+ installé
- UV installé

Configuration

Vérifiez les conditions préalables

Vérifiez que votre version de Python est 3.10 ou supérieure

```
python3 --version
```

Vérifiez l'installation UV

```
uv --version
```

Vérifier la AWS CLI configuration

```
aws configure list
```

Définir des autorisations IAM

Vous aurez besoin des autorisations IAM pour envoyer des requêtes en lecture seule aux AWS services et interagir avec le serveur MCP. Vous pouvez soit tirer parti des politiques AWS gérées, soit utiliser une seule politique personnalisée pour un contrôle plus précis.

Option 1 : politique gérée et personnalisée combinée

1. Joignez la politique AWS gérée ReadOnlyAccess pour un accès en lecture seule à tous les services AWS
2. Créez et joignez une politique personnalisée supplémentaire pour les autorisations MCP (voir les autorisations MCP suivantes au format JSON)

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MCPServerAccess",
      "Effect": "Allow",
      "Action": [
        "ecs-mcp:InvokeReadOnlyTools",
        "ecs-mcp:UseMcp"
      ],
      "Resource": "*"
    }
  ]
}
```

Option 2 : politique personnalisée unique (contrôle le plus granulaire)

Vous pouvez également créer et joindre une seule politique JSON personnalisée qui inclut à la fois les autorisations de AWS service et les autorisations MCP :

```
{
  "Version": "2012-10-17",
  "Statement": [
    {
      "Sid": "MCPServerAccess",
      "Effect": "Allow",
      "Action": [
        "ecs-mcp:InvokeReadOnlyTools",
        "ecs-mcp:UseMcp"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ECSReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
```

```
        "ecs:ListAccountSettings",
        "ecs:ListClusters",
        "ecs:ListContainerInstances",
        "ecs:ListTaskDefinitions",
        "ecs:ListServices",
        "ecs:ListServiceDeployments",
        "ecs:ListTasks",
        "ecs:DescribeClusters",
        "ecs:DescribeCapacityProviders",
        "ecs:DescribeContainerInstances",
        "ecs:DescribeTaskDefinition",
        "ecs:DescribeServices",
        "ecs:DescribeServiceDeployments",
        "ecs:DescribeServiceRevisions",
        "ecs:DescribeTaskSets",
        "ecs:DescribeTasks"
      ],
      "Resource": "*"
    },
    {
      "Sid": "CloudWatchLogsReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "logs:DescribeLogGroups",
        "logs:DescribeLogStreams",
        "logs:GetLogEvents",
        "logs:FilterLogEvents"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ELBReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "elasticloadbalancing:DescribeLoadBalancers",
        "elasticloadbalancing:DescribeTargetGroups",
        "elasticloadbalancing:DescribeListeners"
      ],
      "Resource": "*"
    },
    {
      "Sid": "EC2ReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
```

```
        "ec2:DescribeVpcs",
        "ec2:DescribeSubnets",
        "ec2:DescribeSecurityGroups",
        "ec2:DescribeRouteTables",
        "ec2:DescribeNetworkInterfaces"
      ],
      "Resource": "*"
    },
    {
      "Sid": "ECRReadOnlyAccess",
      "Effect": "Allow",
      "Action": [
        "ecr:DescribeRepositories",
        "ecr:DescribeImages"
      ],
      "Resource": "*"
    },
    {
      "Sid": "IAMReadOnlyAccessForSecurityAnalysis",
      "Effect": "Allow",
      "Action": [
        "iam:GetRole",
        "iam:ListAttachedRolePolicies",
        "iam:GetPolicy",
        "iam:GetPolicyVersion",
        "iam:ListRolePolicies",
        "iam:GetRolePolicy"
      ],
      "Resource": "*"
    }
  ]
}
```

Choisissez et configurez un assistant IA

Installez des assistants MCP-compatible IA ou tout autre MCP-compatible outil. Par exemple, vous pouvez utiliser Kiro, Cline, Cursor ou Claude Code. Configurez ensuite votre assistant de code AI pour qu'il utilise le serveur Amazon ECS MCP via MCP Proxy for AWS, qui est requis pour un accès sécurisé et authentifié au serveur Amazon ECS MCP. Le proxy agit comme un pont côté client et gère l'authentification AWS SIGv4 à l'aide de vos informations d'identification locales. AWS L'exemple ci-dessous utilise l'interface de ligne de commande Kiro. Cliquez sur ce lien pour en savoir plus sur la configuration de MCP dans Kiro.

Localisez le fichier de configuration MCP

- macOS/Linux:

```
~/.kiro/settings/mcp.json
```

- Windows:

```
%USERPROFILE%\.kiro\settings\mcp.json
```

Créez le fichier de configuration s'il n'existe pas.

Ajouter une configuration de serveur MCP

Assurez-vous de remplacer l'espace réservé region (`{region}`) par la région de votre choix (par exemple,`us-west-2`). Reportez-vous au the section called “Conteneurs Linux sur AWS Fargate” pour obtenir la liste complète des régions. Veillez également à remplacer l'`{profile}`espace réservé par le nom de votre AWS CLI profil, par ex. `default`

Pour Mac/Linux :

```
{
  "mcpServers": {
    "ecs-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://ecs-mcp.{region}.api.aws/mcp",
        "--service",
        "ecs-mcp",
        "--profile",
        "{profile}",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Pour Windows:

```
{
  "mcpServers": {
    "ecs-mcp": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "--from",
        "mcp-proxy-for-aws@latest",
        "mcp-proxy-for-aws.exe",
        "https://ecs-mcp.{region}.api.aws/mcp",
        "--service",
        "ecs-mcp",
        "--profile",
        "{profile}",
        "--region",
        "{region}"
      ]
    }
  }
}
```

Vérifier la configuration

Redémarrez Kiro CLI`kiro-cli`, vérifiez que le serveur MCP est chargé et vérifiez `/mcp` les outils disponibles. `/tools`

Vérification de votre configuration

Connexion de test

Posez une question simple à votre assistant IA pour vérifier la connexion :

```
List all ECS clusters in my AWS account
```

Vous devriez voir la liste de vos clusters Amazon ECS.

Discutez avec votre assistant IA qui utilise le serveur Amazon ECS MCP

Exemple 1 : Surveiller les déploiements

```
Check deployment status for my web-service in production-cluster
Show me the ALB URL for my deployed service
Get service events for the last hour
```

Exemple 2 : Étudier l'état du conteneur

```
Show me all tasks that failed in the last 2 hours
Why are my containers failing health checks?
Display container logs for my api-service
```

Exemple 3 : résolution des problèmes

```
Analyze task failures in my production cluster
Check for image pull errors in the last 30 minutes
Why is my task definition stuck in DELETE_IN_PROGRESS state?
```

Exemple 4 : Inspecter les configurations

```
Show me the network configuration for my web-service
What security groups are attached to my service?
List all VPC and subnet details for my ECS service
```

Exemple 5 : évaluer la posture de sécurité

```
Are there any security issues with my production cluster?
Analyze IAM roles and policies for my web-service
Check for overly permissive security groups in my ECS cluster
```

Configurations courantes et meilleures pratiques

Plusieurs AWS profils

Si vous travaillez avec plusieurs AWS comptes, créez des configurations de serveur MCP distinctes.

Pour Mac/Linux :

```
{
  "mcpServers": {
```

```
    "ecs-mcp-prod": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://ecs-mcp.{region}.api.aws/mcp",
        "--service",
        "ecs-mcp",
        "--profile",
        "production",
        "--region",
        "us-west-2"
      ]
    },
    "ecs-mcp-dev": {
      "disabled": false,
      "type": "stdio",
      "command": "uvx",
      "args": [
        "mcp-proxy-for-aws@latest",
        "https://ecs-mcp.{region}.api.aws/mcp",
        "--service",
        "ecs-mcp",
        "--profile",
        "development",
        "--region",
        "us-east-1"
      ]
    }
  }
}
```

Bonnes pratiques de sécurité

Ne transmettez pas de secrets ou d'informations sensibles via les mécanismes de saisie autorisés :

- N'incluez pas de secrets ou d'informations d'identification dans les fichiers de configuration
- Ne transmettez pas d'informations sensibles directement dans les instructions au modèle
- N'incluez pas de secrets dans les définitions de tâches ou les configurations de service
- Évitez de consigner des informations sensibles dans les journaux des applications
- Utilisez ou Parameter Store pour stocker des informations sensibles

Configurations des outils

Pour une liste complète des outils et des configurations, consultezConfigurations de l'outil serveur Amazon ECS MCP.

Configurations de l'outil serveur Amazon ECS MCP

Ce guide présente toutes les configurations disponibles pour le serveur Amazon ECS MCP entièrement géré.

> **Note**
>
> Le serveur Amazon ECS MCP est en version préliminaire et est sujet à modification.

Variables d’environnement

AWS_PROFILE (facultatif)

AWS nom du profil d'informations d'identification à utiliser ; peut être remplacé par l'argument de `--profile` ligne de commande.

Exemple : `export AWS_PROFILE=dev`

AWS_REGION (facultatif)

AWS région pour la signature SIGv4 ; la valeur par défaut est `us-west-2` si elle n'est pas définie.

Exemple : `export AWS_REGION=us-west-2`

Arguments

URL du point de terminaison SIGv4 MCP (obligatoire)

URL du point de terminaison MCP auquel se connecter.

Exemple : `https://ecs-mcp.us-west-2.api.aws/mcp`

--service (facultatif)

AWS nom de service pour la signature SIGv4 ; détecté automatiquement à partir du nom d'hôte du terminal s'il n'est pas fourni.

Exemple : `--service ecs-mcp`

--profile (facultatif)

AWS profil d'informations d'identification à utiliser. La valeur par défaut est la variable d'`AWS_PROFILE`environnement si elle n'est pas spécifiée.

Exemple : `--profile dev`

--région

AWS région à utiliser. Utilise une variable d'`AWS_REGION`environnement si elle n'est pas définie, la valeur par défaut est. `us-east-1`

Exemple : `--region us-west-2`

--lecture seule (facultatif)

Désactivez les outils qui peuvent nécessiter des autorisations d'écriture (les outils qui ne nécessitent PAS d'autorisations d'écriture sont annotés`readOnlyHint=true`). Par défaut, tous les outils sont activés.

Exemple : `--read-only`

Pour plus d'options de configuration, consultez la section Paramètres de configuration.

Outils

Le serveur MCP Amazon ECS expose les outils MCP suivants.

Outils en lecture seule

get_deployment_status

L'outil vérifie l'état de votre déploiement Amazon ECS pour un cluster et un service Amazon ECS spécifiques.

- Actions IAM requises :
  - `ecs:DescribeServices`
  - `elasticloadbalancing:DescribeTargetGroups`
  - `elasticloadbalancing:DescribeLoadBalancers`

- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS
  - `service_name`(chaîne) : nom du service Amazon ECS
- Réponse : L'outil renvoie l'état du déploiement et les détails associés au déploiement.

fetch_service_events

Cet outil extrait les événements du service Amazon ECS à des fins de diagnostic avec des fenêtres temporelles personnalisables.

- Actions IAM requises :
  - `ecs:DescribeServices`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS
  - `service_name`(chaîne) : nom du service Amazon ECS
- Paramètres facultatifs :
  - `time_window`(entier) : Vous pouvez mentionner une fenêtre de temps en secondes (la valeur par défaut est de 1 heure ou 3600 secondes)
  - `start_time`(chaîne) : heure de début personnalisée au format ISO
  - `end_time`(chaîne) : heure de fin personnalisée au format ISO
- Réponse : L'outil renvoie la liste des événements du service avec un résumé.

fetch_task_failures

Cet outil vous permet de récupérer et d'analyser les échecs des tâches Amazon ECS à l'aide de résumés. À l'aide de cet outil, vous pouvez identifier tous les modèles d'échec des tâches.

- Actions IAM requises :
  - `ecs:ListTasks`
  - `ecs:DescribeTasks`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS où se trouve votre tâche
- Paramètres facultatifs :
  - `service_name`(chaîne) : Vous pouvez filtrer par service spécifique
  - `time_window`(entier) : Vous pouvez mentionner une fenêtre de temps en secondes (la valeur par défaut est de 1 heure ou 3600 secondes)

- `start_time`(chaîne) : heure de début personnalisée au format ISO
- `end_time`(chaîne) : heure de fin personnalisée au format ISO

- Réponse : L'outil revient TaskFailureResult avec les tâches ayant échoué et un résumé

fetch_task_logs

Cet outil extrait CloudWatch les journaux des tâches Amazon ECS avec des options de plage de temps flexibles. Vous utilisez cet outil pour résoudre les problèmes d'exécution.

- Actions IAM requises :
  - `ecs:DescribeServices`
  - `ecs:DescribeTaskDefinition`
  - `logs:FilterLogEvents`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS
  - `service_name`(chaîne) : nom du service Amazon ECS
- Paramètres facultatifs :
  - `time_window`(entier) : Vous pouvez mentionner une fenêtre de temps en secondes (la valeur par défaut est de 1 heure ou 3600 secondes)
  - `start_time`(chaîne) : heure de début personnalisée au format ISO
  - `end_time`(chaîne) : heure de fin personnalisée au format ISO
  - `log_group_name`(chaîne) : Spécifiez le groupe de CloudWatch journaux
- Réponse : L'outil renvoie TaskLogsResult des entrées de journal

get_task_definition_deletion_blockers

Cet outil identifie les dépendances qui peuvent empêcher la suppression des définitions de tâches. L'outil est approprié lorsque vous exécutez des opérations de nettoyage et que vous comprenez ce qui empêche la suppression des ressources Amazon ECS.

- Actions IAM requises :
  - `ecs:ListClusters`
  - `ecs:ListServices`
  - `ecs:DescribeServices`
  - `ecs:ListTasks`
  - `ecs:DescribeTasks`

- Paramètres requis :
  - `task_definition_arn`(chaîne) : définition de tâche Amazon Resource Name (ARN) à analyser
- Réponse : L'outil renvoie les bloqueurs et l'état de suppression

detect_image_pull_failures

Cet outil vous permet de détecter et de classer les échecs d'extraction d'images de conteneurs.

- Actions IAM requises :
  - `ecs:ListTasks`
  - `ecs:DescribeTasks`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS
- Paramètres facultatifs :
  - `service_name`(chaîne) : Vous pouvez filtrer par service spécifique
  - `time_window`(entier) : Vous pouvez mentionner une fenêtre de temps en secondes (la valeur par défaut est de 1 heure ou 3600 secondes)
  - `start_time`(chaîne) : heure de début personnalisée au format ISO
  - `end_time`(chaîne) : heure de fin personnalisée au format ISO
- Réponse : L'outil renvoie les échecs et le résumé

fetch_network_configuration

Cet outil récupère les détails de configuration du réseau de service Amazon ECS. Vous pouvez utiliser cet outil pour comprendre la configuration d'Amazon VPC, de sous-réseau et de groupe de sécurité de votre configuration Amazon ECS.

- Actions IAM requises :
  - `ecs:DescribeServices`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS
  - `service_name`(chaîne) : nom du service Amazon ECS
- Réponse : L'outil renvoie une liste contenant les détails du réseau NetworkConfigurationResult

obtenir_recommandations_de sécurité

Cet outil collecte les données de configuration de sécurité du cluster Amazon ECS à des fins d'analyse AI-powered de sécurité. Il regroupe les rôles et les politiques IAM, les paramètres réseau (groupes de sécurité, sous-réseaux, adresse IP publique), les paramètres des conteneurs de définition des tâches et la configuration au niveau du cluster. Votre assistant d'IA compare ces données aux meilleures pratiques AWS de sécurité et génère un rapport de résultats hiérarchisé avec des étapes de correction.

- Actions IAM requises :
  - `ecs:DescribeClusters`
  - `ecs:ListServices`
  - `ecs:DescribeServices`
  - `ecs:DescribeTaskDefinition`
  - `ec2:DescribeSecurityGroups`
  - `ec2:DescribeSubnets`
  - `iam:GetRole`
  - `iam:ListAttachedRolePolicies`
  - `iam:GetPolicy`
  - `iam:GetPolicyVersion`
  - `iam:ListRolePolicies`
  - `iam:GetRolePolicy`
- Paramètres requis :
  - `cluster_name`(chaîne) : nom du cluster Amazon ECS à analyser
- Paramètres facultatifs :
  - `service_name`(string) : analyse de la portée d'un service unique. En cas d'omission, tous les services du cluster sont analysés.
  - `max_services`(entier) : nombre maximum de services à analyser (par défaut 25, plafonné à 50)
- Réponse : L'outil renvoie la configuration du cluster, les configurations de service, les paramètres de définition des tâches, les détails de sécurité du réseau, les données de rôle et de politique IAM, ainsi que tous les avertissements de collecte. Votre assistant IA utilise ces données pour générer un rapport sur les résultats de sécurité classés par ordre de priorité avec des conseils de correction.

fetch_action_logs

Utilisez cet outil pour récupérer les journaux d'action Amazon ECS et résoudre les problèmes liés aux opérations liées au cycle de vie des conteneurs. Grâce à Action Logs, vous bénéficiez d'une meilleure visibilité sur les opérations côté service qu'Amazon ECS effectue pour votre compte, notamment les pièces jointes ENI, les extractions d'images, les inscriptions à l'équilibreur de charge, les nouvelles tentatives de limitation et les erreurs d'autorisation. Utilisez cet outil lorsque les tâches ne démarrent pas, que les déploiements sont bloqués ou que vous avez besoin de comprendre ce qu'Amazon ECS a fait au cours d'une opération. L'outil renvoie les entrées du journal des actions si les journaux des actions sont activés pour le cluster, ou un message d'état si les journaux des actions ne sont pas configurés.

- Actions IAM requises :
  - `ecs:DescribeClusters`
  - `logs:DescribeDeliverySources`
  - `logs:DescribeDeliveries`
  - `logs:DescribeDeliveryDestinations`
  - `logs:FilterLogEvents`
- Paramètres requis :
  - `cluster_name`: nom ou ARN du cluster Amazon ECS.
- Paramètres facultatifs :
  - `service_name`: nom du service Amazon ECS selon lequel filtrer les journaux d'actions.
  - `task_id`: ARN ou ID de tâche selon lequel filtrer les journaux d'actions.
  - `deployment_id`: ID de déploiement selon lequel filtrer les journaux d'actions.
  - `log_level`: le filtre de niveau de journalisation. Les valeurs valides sont ERROR, WARN ou INFO.
  - `time_window`: fenêtre temporelle en secondes. La valeur par défaut est de 3 600 secondes.
  - `start_time`: heure de début personnalisée au format ISO.
  - `end_time`: heure de fin personnalisée au format ISO.
- Réponse :

  L'outil renvoie l'état de découverte des journaux d'action pour le cluster. Si les journaux d'actions sont activés avec une destination CloudWatch Logs, l'outil renvoie également les entrées de journal correspondantes. Chaque entrée comprend un horodatage, un niveau de journalisation, le nom de l'événement, l'ARN de la ressource et les détails de l'événement. Les

résultats sont limités à 1 000 entrées. Pour récupérer des données plus ciblées, utilisez des fenêtres temporelles plus étroites ou des filtres supplémentaires.

Outils de connaissances

Ces outils permettent d'accéder à une AWS documentation actualisée. Ils sont alimentés par le serveur AWS Knowledge MCP et ne nécessitent aucune autorisation IAM de la part de l'appelant : le service Amazon ECS MCP se connecte au Knowledge Endpoint en votre nom.

aws_knowledge_aws___search_documentation

Cet outil recherche dans AWS la documentation les pages pertinentes correspondant à une requête.

- Actions IAM requises : Aucune
- Paramètres requis :
  - `search_phrase`(chaîne) : requête de recherche pour AWS la documentation
- Paramètres facultatifs :
  - `topics`(tableau) : Jusqu'à 3 sujets dans lesquels effectuer une recherche (par défaut`["general"]`)
  - `limit`(entier) : nombre maximum de résultats par sujet (par défaut 5)
- Réponse : L'outil renvoie les pages de documentation correspondantes avec des titres, des URL et des extraits de contexte.

aws_knowledge_aws___read_documentation

Cet outil lit le contenu des pages de AWS documentation et les renvoie au format Markdown.

- Actions IAM requises : Aucune
- Paramètres requis :
  - `requests`(tableau) : tableau d'objets de requête. Chaque objet contient :
    - `url`(chaîne, obligatoire) : URL de la page de documentation à lire
    - `max_length`(entier, facultatif) : nombre maximum de caractères à renvoyer (par défaut 10000)
    - `start_index`(entier, facultatif) : position du caractère à partir de laquelle commencer la lecture (0 par défaut)
- Réponse : L'outil renvoie le contenu de la page de documentation au format Markdown. Les pages longues sont renvoyées par morceaux avec l'aide à la navigation.

Dépannage d'Amazon ECS

Vous devrez peut-être résoudre des problèmes liés à vos équilibreurs de charge, tâches, services ou instances de conteneur. Ce chapitre vous aide à trouver les informations de diagnostic à partir de l'agent de conteneur Amazon ECS, du démon Docker sur l'instance de conteneur et du journal des événements de service de la console Amazon ECS.

Pour plus d’informations sur les tâches arrêtées, consultez l’une des rubriques suivantes.

Action	En savoir plus
Résoudre les erreurs liées aux tâches arrêtées.	Affichage des erreurs liées aux tâches Amazon ECS arrêtées
Afficher les erreurs liées aux tâches arrêtées.	Résolution des erreurs liées aux tâches Amazon ECS arrêtées
Vérifier les codes d’erreur des tâches arrêtées.	Messages d’erreurs liées aux tâches Amazon ECS arrêtées
Révision CannotPullContainer erreurs de tâche.	CannotPullContainer erreurs de tâches dans Amazon ECS
Afficher les requêtes de rôle IAM de tâche.	Affichage des requêtes de rôle IAM pour des tâches Amazon ECS
Résolvez les problèmes à l'aide d'événements de tâche.	Capture d'événements Amazon ECS dans la console

Pour plus d’informations sur les erreurs de service, consultez l’une des rubriques suivantes.

Action	En savoir plus
Afficher les messages d'événements de service.	Affichage des messages d'événement d'un service Amazon ECS
Vérifier les messages d'événements de service.	Messages d'événement de service Amazon ECS
Vérifier les problèmes liés à l'équilibreur de charge.	Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS
Vérifier les problèmes liés à l'autoscaling du service.	Résolution des problèmes d'autoscaling du service dans Amazon ECS

Pour plus d'informations sur les erreurs de définition de tâche, consultez l'une des rubriques suivantes.

Action	En savoir plus
Résoudre l'erreur de mémoire liée à la de définition de tâche.	Résolution des erreurs d'UC ou de mémoire non valides liées à la définition des tâches Amazon ECS

Pour plus d'informations sur les erreurs d'agent Amazon ECS, consultez l'une des rubriques suivantes.

Action	En savoir plus
Afficher le journal d'agent de conteneur Amazon ECS	Affichage du journal d'agent de conteneur Amazon ECS

Action	En savoir plus
Découvrir comment collecter les journaux Amazon ECS.	Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS
Récupérer les informations de diagnostic avec l'agent Amazon ECS.	Récupération des informations de diagnostic d'Amazon ECS avec l'introspection d'agent

Pour plus d'informations sur les erreurs Docker, consultez l'une des rubriques suivantes.

Action	En savoir plus
Utiliser les diagnostics Docker.	Diagnostic Docker dans Amazon ECS
Activer le mode de débogage Docker.	Configuration de la sortie détaillée du démon Docker dans Amazon ECS
Résoudre l'erreur 500 de l'API Docker.	Résoudre les problèmes liés au Docker `Erreur d'API (500) : devmapper` dans Amazon ECS

Pour plus d'informations sur les erreurs d'ECS Exec et d'Amazon ECS Anywhere, consultez l'une des rubriques suivantes.

Action	En savoir plus
Résoudre les problèmes liés à ECS Exec.	Résolution des problèmes liés à Amazon ECS Exec
Résoudre les problèmes liés à Amazon ECS Anywhere.	Résolution des problèmes liés à Amazon ECS Anywhere

Pour plus d'informations sur les problèmes liés à l'association de volumes Amazon EBS à des tâches Amazon ECS, consultez l'un des points suivants :

Action	En savoir plus	
Résolution des problèmes liés à l'association de volumes Amazon EBS à des tâches Amazon ECS.	Résolution des problèmes liés à l'attachement de volumes Amazon EBS aux tâches Amazon ECS	
Raisons du statut des volumes Amazon EBS associés aux tâches Amazon ECS.	Raisons du statut de l'attachement d'un volume Amazon EBS aux tâches Amazon ECS	

Pour plus d'informations sur les problèmes liés à l'utilisation du partage AWS Cloud Map espaces de noms avec Amazon ECS Service Connect, consultez l'une des rubriques suivantes :

Action	En savoir plus	
Résolution des problèmes liés à Amazon ECS Service Connect avec un service partagé AWS Cloud Map espaces de noms.	Résolution des problèmes liés à Amazon ECS Service Connect avec un service partagé AWS Cloud Map espaces de noms	

Pour plus d'informations sur les problèmes de limitation, consultez l'une des rubriques suivantes :

Action	En savoir plus	
En savoir plus sur les quotas de limitation de Fargate.	AWS Fargate limitation des quotas	
Découvrir les pratiques exemplaires en matière de limitation pour Amazon ECS.	Gestion des problèmes de limitation d'Amazon ECS	

Pour plus d'informations sur les erreurs d'API, consultez l'une des rubriques suivantes.

Action	En savoir plus	
Résoudre les erreurs d'API.	Motifs d'échec d'API Amazon ECS	

Pour plus d'informations sur AI-powered pour résoudre les problèmes, consultez les rubriques suivantes :

Action	En savoir plus	
Résolvez les problèmes liés à Amazon Q Developer dans la console.	Résolution des problèmes avec Amazon Q Developer	
Résolvez les problèmes liés aux assistants IA à l'aide du serveur Amazon ECS MCP.	Serveur Amazon ECS MCP	

Résolution des erreurs liées aux tâches Amazon ECS arrêtées

Lorsque votre tâche ne démarre pas, un message d'erreur s'affiche dans la console et dans les paramètres de sortie `describe-tasks` (`stoppedReason` et `stopCode`).

Vous pouvez consulter les tâches arrêtées dans la console pendant une heure. Pour voir les tâches arrêtées, vous devez modifier l'option de filtre. Pour plus d'informations, consultez Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Les pages suivantes fournissent des informations sur les tâches arrêtées.

- Découvrez les modifications apportées aux messages d'erreur relatifs aux tâches arrêtées.

  Mises à jour des messages d'erreurs liées aux tâches Amazon ECS arrêtées
- Afficher vos tâches arrêtées afin d'obtenir des informations sur la cause.

  Affichage des erreurs liées aux tâches Amazon ECS arrêtées

- En savoir plus sur les messages d'erreur liés aux tâches arrêtées et les causes possibles de ces erreurs.

  Messages d'erreurs liées aux tâches Amazon ECS arrêtées
- Découvrir comment vérifier la connectivité des tâches arrêtées et corriger les erreurs.

  Vérification de la connectivité de la tâche Amazon ECS arrêtée

Mises à jour des messages d'erreurs liées aux tâches Amazon ECS arrêtées

À compter du 14 juin 2024, l'équipe Amazon ECS modifiera les messages d'erreur relatifs aux tâches arrêtées, comme décrit dans les tableaux suivants. Les `stopCode` ne changeront pas. Si vos applications sont basées sur une correspondance exacte des chaînes de messages d'erreur, vous devez les mettre à jour avec les nouvelles chaînes. Pour obtenir de l'aide en cas de questions ou de problèmes, contactez AWS Support.

> **Note**
>
> Nous vous recommandons de ne pas vous fier aux messages d'erreur pour votre automatisation, car ils sont susceptibles de changer.

CannotPullContainerError

Ancien message d'erreur	Nouveau messages d'erreur	
CannotPullContainerError: Réponse d'erreur du démon : accès au pull refusé pour *repository* , le référentiel n'existe pas ou peut nécessite r une « connexion docker » : refusé : Utilisateur : *roleARN*	• CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre. Réponse d'erreur du démon : accès au pull refusé pour *repositor y* , le référentiel n'existe	

Ancien message d'erreur	Nouveau messages d'erreur	
	pas ou peut nécessiter une « connexion docker » : refusé : Utilisateur : *roleARN* n'est pas autorisé à exécuter : ecr :BatchGet Image sur la ressource : *image* car aucune politique basée sur l'identité n'autoris e l'ecr :BatchGetImage action. • CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez si l'image existe. Réponse d'erreur du démon : accès au pull refusé pour *repository* , le référenti el n'existe pas ou peut nécessiter une « connexion docker » : refusé : l'accès demandé à la ressource est refusé.	
CannotPullContainerError: Réponse d'erreur du démon : Get *imageURI*: net/http: demande annulée en attente de connexion	CannotPullContainerError: La tâche ne peut pas extraire l'image. Vérifiez la configura tion de votre réseau. Réponse d'erreur du démon : Get *image*: net/http: demande annulée en attente de connexion (Client.Timeout dépassé en attendant les en-têtes)	

Ancien message d'erreur	Nouveau messages d'erreur	
CannotPullContainerError: ref pull a été réessayé 5 fois : échec de la copie : httpReadSeeker: échec de l'ouverture : échec de l'exécution de la demande : Get *registry-uri* <ip>: composez le TCP : <port> i/o délai d'expiration	CannotPullContainerError: La tâche ne peut pas être retirée *image-uri* à partir du registre *registry-uri* . Il existe un problème de connexion entre la tâche et le registre. Vérifiez la configuration de votre réseau de tâches. : échec de la copie : httpReadSeeker: échec de l'ouverture : échec de l'exécution de la demande : Get *registry-uri* <ip>: composez le TCP : <port> i/o délai d'expiration	

ResourceNotFoundException

Ancien message d'erreur	Nouveau messages d'erreur	
Récupération de données secrètes depuis AWS Secrets Manager dans *region* région : secrète *sercretARN* : ResourceNotFoundException: Le gestionnaire de secrets ne trouve pas le secret spécifié.	ResourceNotFoundException: La tâche ne peut pas récupérer le secret avec l'ARN '*sercretARN* 'à partir de AWS Secrets Manager. Vérifiez si le secret existe dans la région spécifiée . ResourceNotFoundException: Récupération de données secrètes depuis AWS Secrets Manager dans la région *region*: secret *sercretARN* : ResourceNotFoundException: Le	

Ancien message d’erreur	Nouveau messages d’erreur	
	gestionnaire de secrets ne trouve pas le secret spécifié.	

ResourceInitializationError

Ancien message d’erreur	Nouveau messages d’erreur	
ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : échec de la récupération des ressource s d'exécution : impossible de récupérer l'authentification du registre ECR : l'appel de service a été retenté 3 fois : RequestError: échec de l'envoi de la demande en raison de : « Post »https://api.ecr.us-east-1.amazonaws.com/<por t>« : composez le numéro tcp <ip>: : i/o délai d'attente . Veuillez vérifier la configura tion réseau de votre tâche.	ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : la tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR : il y a un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration de votre réseau de tâches. RequestEr ror: échec de l'envoi de la demande en raison de : « Post »https://api.ecr.us-east-1.amazonaws.com<port >« : composez le numéro tcp <ip>: : i/o délai d'expiration	
ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : échec de la récupération des ressource s d'exécution : impossible de récupérer les secrets depuis SSM : l'appel de service a été retenté 5 fois : RequestCa	ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : impossible de récupérer les secrets depuis ssm : la tâche ne peut pas extraire les secrets de AWS Systems Manager. Il existe un problème de	

Ancien message d’erreur	Nouveau messages d’erreur	
nceled: le contexte de la demande a été annulé en raison d'un dépassement de la date limite du contexte	connexion entre la tâche et le magasin de paramètres AWS Systems Manager . Vérifiez la configuration de votre réseau de tâches. RequestCanceled: le contexte de la demande a été annulé en raison d'un dépassement de la date limite du contexte	
ResourceInitializationError: échec du téléchargement des fichiers env : commande de téléchargement de fichier : flux d'erreur non vide : RequestCa nceled: le contexte de la demande a été annulé en raison d'un dépassement de la date limite du contexte	ResourceInitializationError: échec du téléchargement des fichiers d'environnement : la tâche ne peut pas télécharg er les fichiers de variables d'environnement depuis Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration de votre réseau de tâches. L'appel de service a été relancé 5 fois : RequestCa nceled: le contexte de la demande a été annulé en raison d'un dépassement de la date limite du contexte	

Ancien message d’erreur	Nouveau messages d’erreur	
ResourceInitializationError : échec de la validation du logger args : :signal:killed	ResourceInitializationError : échec de la validation des arguments du logger : la tâche ne trouve pas l'Amazon CloudWatch groupe de journaux défini dans la définition de la tâche. Il y a un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau. : signal : supprimé	
ResourceInitializationError : impossible d'extraire les secrets ou l'authentification du registre : échec de la commande pull : : signal : tué	ResourceInitializationError : impossible d'extraire des secrets ou d'authentifier le registre : vérifiez la configura tion réseau de vos tâches. : signal : tué	

Affichage des erreurs liées aux tâches Amazon ECS arrêtées

Si vous avez des difficultés à démarrer une tâche, cette dernière peut avoir été arrêtée en raison d'une application ou d'erreurs de configuration. Par exemple, vous exécutez la tâche et elle affiche un statut `PENDING`, puis disparaît.

Si votre tâche a été créée par un service Amazon ECS, les actions entreprises par Amazon ECS pour gérer le service sont publiées dans les événements du service. Vous pouvez consulter les événements dans le Console de gestion AWS, AWS CLI, AWS Des kits de développement logiciel, l'API Amazon ECS ou des outils qui utilisent les kits de développement logiciel et l'API. Ces événements incluent l'arrêt et le remplacement d'une tâche par Amazon ECS parce que les conteneurs de la tâche ont cessé de s'exécuter ou parce qu'un trop grand nombre de surveillances de l'état effectuées par Elastic Load Balancing ont échoué.

Si votre tâche a été exécutée sur une instance de conteneur sur Amazon EC2 ou sur des ordinateurs externes, vous pouvez également consulter les journaux de l'environnement d'exécution du conteneur et de l'agent Amazon ECS. Ces journaux se trouvent sur l'instance Amazon EC2 hôte ou sur un ordinateur externe. Pour plus d'informations, consultez Affichage du journal d'agent de conteneur Amazon ECS.

Procédure

Console

Console de gestion AWS

Les étapes suivantes peuvent être utilisées pour rechercher des erreurs dans les tâches arrêtées à l'aide de la console. Pour voir les tâches arrêtées, vous devez modifier l'option de filtre.

Les tâches arrêtées n'apparaissent dans la console que pendant une heure.

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Sur le Cluster : ***name*** page, choisissez Tâches onglet.
5. Configurez le filtre pour afficher les tâches arrêtées. Pour Filtrer selon le statut souhaité, choisissez Arrêté.

   L'option Arrêté affiche vos tâches arrêtées et l'option Tout statut souhaité affiche toutes vos tâches.
6. Choisissez la tâche arrêtée à inspecter.
7. Dans la ligne correspondant à votre tâche arrêtée, dans la colonne Dernier statut, choisissez Arrêté.

   Une fenêtre contextuelle affiche le motif de l'arrêt.

AWS CLI

1. Répertoriez les tâches arrêtées d'un cluster. La sortie contient l'Amazon Resource Name (ARN) de la tâche, dont vous avez besoin pour décrire la tâche.

```
aws ecs list-tasks \
```

```
    --cluster cluster_name \
    --desired-status STOPPED \
    --region region
```

2. Décrivez la tâche arrêtée pour récupérer les informations. Pour plus d'informations, voir décrire les tâches dans le AWS Command Line Interface Référence.

```
aws ecs describe-tasks \
    --cluster cluster_name \
    --tasks arn:aws:ecs:region:account_id:task/cluster_name/task_ID \
    --region region
```

Utilisez les paramètres de sortie suivants.

- `stopCode` - Le code d'arrêt indique pourquoi une tâche a été arrêtée, par exemple ResourceInitializationError
- `StoppedReason` : la raison pour laquelle la tâche s'est arrêtée.
- `reason` (dans la structure `containers`) : la raison fournit des détails supplémentaires sur le conteneur arrêté.

Étapes suivantes

Afficher vos tâches arrêtées afin d'obtenir des informations sur la cause. Pour plus d'informations, consultez Messages d'erreurs liées aux tâches Amazon ECS arrêtées.

Messages d'erreurs liées aux tâches Amazon ECS arrêtées

Voici les messages d'erreur possibles que vous pouvez recevoir lorsque votre tâche est arrêtée de façon inattendue.

Pour vérifier la présence d'un message d'erreur dans vos tâches interrompues à l'aide du Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

> **Tip**
>
> Vous pouvez utiliser le Serveur Amazon ECS MCP avec des assistants intelligents pour analyser les échecs de tâches et les journaux des conteneurs en langage naturel.

Les codes d'erreur des tâches interrompues sont associés à une catégorie, par exemple « »ResourceInitializationError«. Pour plus d'informations sur chaque catégorie, consultez les rubriques suivantes :

Catégorie	En savoir plus
TaskFailedToStart	Résolution des problèmes d'Amazon ECS TaskFailedToStart erreurs
ResourceInitializationError	Résolution des problèmes d'Amazon ECS ResourceInitializationError erreurs
ResourceNotFoundException	Résolution des problèmes d'Amazon ECS ResourceNotFoundException erreurs
SpotInterruptionError	Résolution des problèmes d'Amazon ECS SpotInterruption erreurs
InternalError	Résolution des problèmes d'Amazon ECS InternalError erreurs
OutOfMemoryError	Résolution des problèmes d'Amazon ECS OutOfMemoryError erreurs
ContainerRuntimeError	Résolution des problèmes d'Amazon ECS ContainerRuntimeError erreurs
ContainerRuntimeTimeoutError	Résolution des problèmes d'Amazon ECS ContainerRuntimeTimeoutError erreurs

Catégorie	En savoir plus	
CannotStartContainerError	Résolution des problèmes d'Amazon ECS CannotStartContainerError erreurs	
CannotStopContainerError	Résolution des problèmes d'Amazon ECS CannotStopContainerError erreurs	
CannotInspectContainerError	Résolution des problèmes d'Amazon ECS CannotInspectContainerError erreurs	
CannotCreateVolumeError	Résolution des problèmes d'Amazon ECS CannotCreateVolumeError erreurs	
CannotPullContainer	CannotPullContainer erreurs de tâches dans Amazon ECS	

Résolution des problèmes d'Amazon ECS TaskFailedToStart erreurs

Vous trouverez ci-dessous quelques messages d'erreur `TaskFailedToStart` et les actions que vous pouvez entreprendre pour corriger ces erreurs.

Pour vérifier la présence d'un message d'erreur dans vos tâches interrompues à l'aide du Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Erreur EC2 inattendue lors de la tentative de création d'une interface réseau avec l'attribution d'adresses IP publiques activée dans le sous-réseau '***identifiant de sous-réseau***

Cela se produit lorsqu'une tâche Fargate utilisant le mode réseau `awsvpc` s'exécute dans un sous-réseau doté d'une adresse IP publique et que le sous-réseau ne possède pas suffisamment d'adresses IP.

Le nombre d'adresses IP disponibles est indiqué sur la page des détails du sous-réseau dans la console Amazon EC2 ou en utilisant `describe-subnets`. Pour plus d'informations, consultez la section Afficher votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

Pour résoudre ce problème, vous pouvez créer un sous-réseau dans lequel exécuter votre tâche.

InternalError: *<raison>*

Cette erreur se produit lorsqu'une pièce jointe ENI est demandée. Amazon EC2 gère de manière asynchrone le provisionnement de l'ENI. Le processus de provisionnement prend du temps. Amazon ECS dispose d'un délai d'expiration au cas où les temps d'attente seraient longs ou les défaillance non signalées. Parfois, l'ENI est provisionnée, mais le rapport arrive à Amazon ECS après l'expiration du délai de défaillance. Dans ce cas, Amazon ECS détecte la défaillance de la tâche signalée avec un ENI en cours d'utilisation.

La définition de tâche sélectionnée n’est pas compatible avec la stratégie de calcul sélectionnée

Cette erreur se produit lorsque vous avez choisi une définition de tâche dont le type de lancement ne correspond pas au type de capacité du cluster. Vous devez sélectionner une définition de tâche qui correspond au fournisseur de capacité attribué à votre cluster.

Impossible de connecter l’interface réseau à l’index des appareils non utilisés

Cette erreur se produit lors de l'utilisation `awsvpc` type de réseau et il n'y en a pas assez CPU/ memory pour la tâche. Tout d’abord, vérifiez que les instances sont bien présentes sur l’UC. Pour plus d’informations, consultez la section Spécifications des types d’instances Amazon EC2 dans les Types d’instances Amazon EC2. Prenez la valeur d’UC de l’instance et multipliez-la par le nombre d’ENI de l’instance. Utilisez cette valeur e dans la définition de tâche.

AGENT

L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellement déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée.

Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.

Résolution des problèmes d'Amazon ECS ResourceInitializationError erreurs

Vous trouverez ci-dessous quelques messages d’erreur `ResourceInitialization` et les actions que vous pouvez entreprendre pour corriger ces erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Erreurs

- La tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration réseau de votre tâche.
- La tâche ne peut pas télécharger les fichiers de variables d'environnement à partir d'Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration réseau de votre tâche.
- La tâche ne peut pas extraire de secrets AWS Systems Manager Magasin de paramètres. Vérifiez votre connexion réseau entre la tâche et AWS Systems Manager.
- La tâche ne peut pas révéler de secrets AWS Secrets Manager. Il existe un problème de connexion entre la tâche et Secrets Manager. Vérifiez la configuration réseau de votre tâche.
- La tâche ne peut pas extraire le secret depuis Secrets Manager. La tâche ne peut pas récupérer le secret avec l'ARN 'secretARN'de Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.
- échec de la commande d'extraction : impossible d'extraire les secrets ou l'authentification du registre Vérifiez la configuration réseau de votre tâche.
- La tâche ne trouve pas l'Amazon CloudWatch groupe de journaux défini dans la définition de la tâche. Il y a un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau.
- échec de l'initialisation du pilote de journalisation
- échec de l'invocation des commandes EFS utils pour configurer les volumes EFS

La tâche ne peut pas extraire l'authentification du registre depuis Amazon ECR. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration réseau de votre tâche.

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR.

Vérifiez la connexion entre la tâche et Amazon ECR. Pour plus d'informations, consultez Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne peut pas télécharger les fichiers de variables d'environnement à partir d'Amazon S3. Il existe un problème de connexion entre la tâche et Amazon S3. Vérifiez la configuration réseau de votre tâche.

Cette erreur se produit lorsque votre tâche ne parvient pas à télécharger votre fichier d'environnement depuis Amazon S3.

Vérifiez la connexion entre la tâche et le point de terminaison de VPC Amazon S3. Pour plus d'informations, consultez Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne peut pas extraire de secrets AWS Systems Manager Magasin de paramètres. Vérifiez votre connexion réseau entre la tâche et AWS Systems Manager.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de tâche à l'aide des informations d'identification dans Systems Manager.

Vérifiez la connexion entre la tâche et le point de terminaison de VPC Systems Manager. Pour plus d'informations, consultez Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne peut pas révéler de secrets AWS Secrets Manager. Il existe un problème de connexion entre la tâche et Secrets Manager. Vérifiez la configuration réseau de votre tâche.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de tâche à l'aide des informations d'identification dans Secrets Manager.

L'erreur indique qu'il existe un problème de connectivité réseau entre le point de terminaison de VPC Systems Manager et la tâche.

Pour en savoir plus sur la vérification de la connectivité entre la tâche et le point de terminaison, consultez la section Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne peut pas extraire le secret depuis Secrets Manager. La tâche ne peut pas récupérer le secret avec l'ARN '***secretARN***'de Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.

Cette erreur se produit lorsque votre tâche ne parvient pas à extraire l'image définie dans la définition de tâche à l'aide des informations d'identification dans Secrets Manager.

Cela peut être dû à l'une des raisons suivantes.

Cause de l'erreur.	Faites ceci...	
Problème de connectivité réseau entre le point de terminaison de VPC Secrets Manager et la tâche.	Vérifiez la connectivité entre la tâche et le point de terminais on Secrets Manager. Pour plus d'informations, consultez Vérification de la connectiv	

Cause de l’erreur.	Faites ceci...	
Le problème est lié au réseau lorsque l’une des chaînes suivantes apparaît dans le message d’erreur : • établissement de connexion TCP • établissement de connexion UDP • <ip>: <port>: i/o délai d'expiration • net/http: délai d'expiration du handshake TLS • lecture : délai de connexion expiré • Client.Timeout dépassé en attendant les en-têtes • net/http: demande annulée en attente de connexion • signal : supprimé • dépassement du délai du contexte	ité de la tâche Amazon ECS arrêtée.	
Le rôle d’exécution de la tâche défini dans la définition de la tâche ne dispose pas des autorisations nécessaires pour Secrets Manager.	Ajoutez les autorisations requises pour Secrets Manager au rôle d’exécution de tâche. Pour plus d’informations, consultez Autorisations Secrets Manager ou Systems Manager.	

Cause de l’erreur.	Faites ceci...	
L’ARN secret n’existe pas	Vérifiez que l’ARN existe dans Secrets Manager. Pour plus d’informations sur l’afficha ge de vos images, consultez la section Rechercher des secrets dans Secrets Manager du Guide du développeur Secrets Manager.	

échec de la commande d’extraction : impossible d’extraire les secrets ou l’authentification du registre Vérifiez la configuration réseau de votre tâche.

Cette erreur se produit lorsque votre tâche ne parvient pas à se connecter à Amazon ECR, Systems Manager ou Secrets Manager. Cela est dû à une mauvaise configuration de votre réseau.

Pour résoudre ce problème, vérifiez la connectivité entre la tâche et Amazon ECR. Vous devez également vérifier la connectivité entre votre tâche et le service qui stocke votre secret (Systems Manager ou Secrets Manager). Pour plus d’informations, consultez Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne trouve pas l'Amazon CloudWatch groupe de journaux défini dans la définition de la tâche. Il y a un problème de connexion entre la tâche et Amazon CloudWatch. Vérifiez la configuration de votre réseau.

Cette erreur se produit lorsque votre tâche ne parvient pas à trouver le CloudWatch groupe de journaux que vous avez défini dans la définition de la tâche.

L'erreur indique qu'il existe un problème de connectivité réseau entre CloudWatch Le point de terminaison du VPC et la tâche.

Pour en savoir plus sur la vérification de la connectivité entre la tâche et le point de terminaison, consultez la section Vérification de la connectivité de la tâche Amazon ECS arrêtée.

échec de l’initialisation du pilote de journalisation

Cette erreur se produit lorsque votre tâche ne parvient pas à trouver le CloudWatch groupe de journaux que vous avez défini dans la définition de la tâche.

L'erreur indique que CloudWatch le groupe dans la définition de la tâche n'existe pas.

Procédez comme suit pour retrouver les objets manquants CloudWatch.

1. Exécutez la commande suivante pour obtenir les informations de définition de la tâche.

```
aws ecs describe-task-definition \
    --task-definition task-def-name
```

   Examinez le résultat pour chaque conteneur et notez la valeur `awslogs-group`.

```
"logConfiguration": {
                "logDriver": "awslogs",
                "options": {
                    "awslogs-group": "/ecs/example-group",
                    "awslogs-create-group": "true",
                    "awslogs-region": "us-east-1",
                    "awslogs-stream-prefix": "ecs"
                },
```

2. Vérifiez que le groupe existe dans CloudWatch pour plus d'informations, voir Utilisation de groupes de journaux et de flux de journaux dans le Amazon CloudWatch Guide de l'utilisateur des journaux.

   Le problème est soit que le groupe spécifié dans la définition de la tâche est incorrect, soit que le groupe de journaux n'existe pas.

3. Résolvez le problème.

Le problème est…	Faites ceci...
Un groupe de journal incorrect est spécifié dans la définition de tâche.	Mettez à jour la définitio n de tâche pour inclure la configuration du groupe de journaux dans la définitio n du conteneur.Pour plus d'informations sur la mise à jour de la définition de tâche, voir Mise à jour d'une définition de tâche Amazon

Le problème est…	Faites ceci...
	ECS à l’aide de la console ou RegisterTaskDefinition dans le Référence de l'API Amazon Elastic Container Service.
Le groupe de journaux n'existe pas dans CloudWatch	Créez le groupe de journaux. Pour plus d'informations, voir Créez un groupe de journaux dans CloudWatch Journaux dans le Amazon CloudWatch Guide de l'utilisateur des journaux.

échec de l’invocation des commandes EFS utils pour configurer les volumes EFS

Les problèmes suivants peuvent vous empêcher de monter vos volumes Amazon EFS selon vos demandes :

- Le système de fichiers Amazon EFS n’est pas configuré correctement.
- La tâche ne dispose pas des autorisations requises.
- Certains problèmes sont liés à la configuration réseau et VPC.

Pour plus d'informations sur le débogage et la résolution de ce problème, consultez Pourquoi ne puis-je pas monter mes volumes Amazon EFS sur mon AWS Fargate tâches sur AWS Re : POST.

Résolution des problèmes d'Amazon ECS ResourceNotFoundException erreurs

Vous trouverez ci-dessous quelques messages d’erreur `ResourceNotFoundException` et les actions que vous pouvez entreprendre pour corriger ces erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

La tâche ne peut pas récupérer le secret avec l'ARN '*Tarn secret*'de AWS Secrets Manager. Vérifiez si le secret existe dans la région spécifiée.

Cette erreur se produit lorsque la tâche ne parvient pas à extraire le secret depuis Secrets Manager. Cela signifie que le secret spécifié dans la définition de la tâche (et contenu dans le message d'erreur) n'existe pas dans Secrets Manager.

La région figure dans le message d'erreur.

Récupération de données secrètes depuis AWS Secrets Manager dans la région *region*: secret *sercretARN*: ResourceNotFoundException: Le gestionnaire de secrets ne trouve pas le secret spécifié.

Pour en savoir plus sur la recherche de Secrets Manager, consultez la section Rechercher des secrets dans AWS Secrets Manager dans le Guide de l'utilisateur AWS Secrets Manager .

Utilisez le tableau suivant pour déterminer et corriger l'erreur.

Problème	Actions	
Le secret se trouve dans une région différente de celle de la définition de la tâche.	a. Créez le secret dans la même région que la tâche. Pour plus d'informations, voir Créez un AWS Secrets Manager secret. b. Mettez à jour la définition de la tâche avec le nouveau secret. Pour plus d'informations, voir Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console ou RegisterTaskDefinition dans le Référence de l'API Amazon Elastic Container Service.	
L'ARN secret de la définition de tâche est incorrect. Le	Mettez à jour la définition de la tâche avec le secret	

Problème	Actions
secret correct existe dans Secrets Manager.	correct. Pour plus d'informations, voir Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console ou RegisterTaskDefinition dans le Référence de l'API Amazon Elastic Container Service.
Le secret n'existe plus.	a. Créez le secret dans la même région que la tâche. Pour plus d'informations, voir Créez un AWS Secrets Manager secret. b. Mettez à jour la définition de la tâche avec le nouveau secret. Pour plus d'informations, voir Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console ou RegisterTaskDefinition dans le Référence de l'API Amazon Elastic Container Service.

Résolution des problèmes d'Amazon ECS SpotInterruption erreurs

L'erreur `SpotInterruption` a des raisons différentes pour Fargate et EC2.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Fargate

L'erreur `SpotInterruption` se produit lorsqu'il n'y a pas de capacité Fargate Spot ou lorsque Fargate récupère la capacité de Fargate Spot.

Vous pouvez exécuter vos tâches dans plusieurs zones de disponibilité pour augmenter la capacité.

EC2

Cette erreur se produit lorsqu'aucune instance Spot n'est disponible ou lorsqu'EC2 récupère la capacité de l'instance Spot.

Vous pouvez exécuter vos instances dans plusieurs zones de disponibilité pour augmenter la capacité.

Résolution des problèmes d'Amazon ECS InternalError erreurs

S'applique à : Fargate

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

L'erreur `InternalError` se produit lorsque l'agent rencontre une erreur interne inattendue, non liée à l'exécution.

Cette erreur se produit uniquement si vous utilisez la version `1.4` ou ultérieure de la plateforme.

Pour plus d'informations sur la manière de déboguer et de résoudre ce problème, consultez la section Messages d'erreurs liées aux tâches Amazon ECS arrêtées.

Résolution des problèmes d'Amazon ECS OutOfMemoryError erreurs

Voici quelques OutOfMemoryError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

conteneur supprimé en raison de l'utilisation de la mémoire

Cette erreur se produit lorsqu'un conteneur est fermé parce que les processus qu'il contient consomment plus de mémoire que celle allouée dans la définition de la tâche, ou en raison de contraintes liées à l'hôte ou au système d'exploitation.

Résolution des problèmes d'Amazon ECS ContainerRuntimeError erreurs

Voici quelques ContainerRuntimeError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

ContainerRuntimeError

Cette erreur se produit lorsque l'agent reçoit une erreur inattendue de `containerd` pour une opération spécifique à l'exécution. Cette erreur est généralement causée par une défaillance interne de l'agent ou de l'environnement d'exécution `containerd`.

Cette erreur se produit uniquement si vous utilisez la version `1.4.0` ou ultérieure de la plateforme (Linux) ou `1.0.0`ou une version ultérieure (Windows).

Pour plus d'informations sur le débogage et la résolution de ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re : POST.

Résolution des problèmes d'Amazon ECS ContainerRuntimeTimeoutError erreurs

Voici quelques ContainerRuntimeTimeoutError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Impossible de passer à l’exécution ; délai d’attente expiré après une minute ou erreur de délai d’attente Docker.

Cette erreur se produit lorsqu'un conteneur n'a pas pu passer à l'état `RUNNING` ou `STOPPED` dans le délai d'expiration. La raison et la valeur du délai d'expiration sont fournies dans le message d'erreur.

Résolution des problèmes d'Amazon ECS CannotStartContainerError erreurs

Voici quelques CannotStartContainerError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

Impossible d'obtenir l'état du conteneur : ***<raison>***

Cette erreur se produit lorsqu'un conteneur ne peut pas être démarré.

Si votre conteneur tente de dépasser la mémoire spécifiée ici, il est arrêté. Augmentez la mémoire présentée au conteneur. Il s'agit du paramètre `memory` figurant dans la définition de la tâche. Pour plus d'informations, consultez the section called "Mémoire".

Résolution des problèmes d'Amazon ECS CannotStopContainerError erreurs

Voici quelques CannotStopContainerError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

CannotStopContainerError

Cette erreur se produit lorsqu'un conteneur ne peut pas être arrêté.

Pour plus d'informations sur le débogage et la résolution de ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re : POST.

Résolution des problèmes d'Amazon ECS CannotInspectContainerError erreurs

Voici quelques CannotInspectContainerError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

CannotInspectContainerError

Cette erreur se produit lorsque l'agent de conteneur ne peut pas décrire le conteneur via son environnement d'exécution.

Lors de l'utilisation de la version de plateforme `1.3` ou d'une version antérieure, l'agent ECS renvoie le motif de l'erreur à partir de Docker.

Lors de l'utilisation de la version de plateforme `1.4.0` ou ultérieure (Linux) ou de la version de plateforme `1.0.0` ou ultérieure (Windows), l'agent Fargate renvoie le motif de l'erreur depuis `containerd`.

Pour plus d'informations sur le débogage et la résolution de ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re : POST.

Résolution des problèmes d'Amazon ECS CannotCreateVolumeError erreurs

Voici quelques CannotCreateVolumeError les messages d'erreur et les mesures que vous pouvez prendre pour corriger les erreurs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

CannotCreateVolumeError

Cette erreur se produit lorsque l'agent ne parvient pas à créer le montage de volume spécifié dans la définition de tâche.

Cette erreur se produit uniquement si vous utilisez la version `1.4.0` ou ultérieure de la plateforme (Linux) ou `1.0.0`ou une version ultérieure (Windows).

Pour plus d'informations sur le débogage et la résolution de ce problème, consultez Pourquoi ma tâche Amazon ECS est-elle arrêtée sur AWS Re : POST.

CannotPullContainer erreurs de tâches dans Amazon ECS

Les erreurs suivantes indiquent que la tâche n’a pas pu démarrer, car Amazon ECS ne parvient pas à récupérer l’image de conteneur spécifiée.

> **Note**
>
> La version 1.4 de la plateforme Fargate tronque les messages d'erreur longs.

Pour vérifier l'absence de message d'erreur dans vos tâches interrompues, utilisez le Console de gestion AWS, voir Affichage des erreurs liées aux tâches Amazon ECS arrêtées.

> **Tip**
>
> Vous pouvez utiliser le Serveur Amazon ECS MCP avec des assistants IA pour étudier les erreurs d'extraction d'images à l'aide du langage naturel.

Erreurs

- La tâche ne peut pas extraire l’image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre.
- La tâche ne peut pas extraire 'nom de l'image'depuis le référentiel Amazon ECR'URI du référentiel'. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration réseau de votre tâche.
- La tâche ne peut pas extraire l’image. Vérification de la configuration de votre réseau
- CannotPullContainerError: le manifeste pull image a été réessayé 5 fois : échec de la résolution de la référence
- Erreur d'API (500) : Obtenir https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: demande annulée en attente de connexion
- Erreur d'API
- écrire/var/lib/docker/tmp/GetImageBlob111111111: il ne reste plus d'espace sur l'appareil
- ERREUR : toomanyrequests : trop de requêtes ou vous avez atteint votre limite de taux d’extraction.
- Réponse d'erreur du démon : Get url: net/http: demande annulée en attente de connexion
- ref pull a été réessayé 1 fois : échec de la copie : httpReaderSeeker: échec de l'ouverture : code d'état inattendu
- accès en extraction refusé
- échec de la commande d’extraction : panique : erreur d’exécution : adresse mémoire non valide ou déréférencement de pointeur nul
- erreur d'extraction de l'image conf/error extraction de la configuration de l'image
- Contexte annulé

La tâche ne peut pas extraire l’image. Vérifiez que le rôle dispose des autorisations nécessaires pour extraire des images du registre.

Cette erreur indique que la tâche ne peut pas extraire l’image spécifiée dans la définition de la tâche en raison de problèmes d’autorisation.

Pour résoudre ce problème :

1. Vérifiez que l'image existe dans *`repository`*. Pour plus d’informations sur l’affichage de vos images, consultez la section Affichage des détails des images dans Amazon ECR dans le guide de l’utilisateur d’Amazon Elastic Container Registry.

2. Vérifiez que *role-arn* dispose des autorisations appropriées pour extraire l'image.

   Pour plus d'informations sur la mise à jour des rôles, consultez la section Mettre à jour les autorisations d'un rôle dans le Guide d'utilisation Gestion des identités et des accès AWS .

   La tâche utilise l'un des rôles suivants :

   - Pour les tâches utilisant Fargate, il s'agit du rôle d'exécution des tâches. Pour plus d'informations sur les autorisations supplémentaires pour Amazon ECR, consultez la section Tâches Fargate extrayant des images Amazon ECR via les autorisations des points de terminaison d'interface.
   - Pour les tâches avec EC2, il s'agit du rôle d'instance de conteneur. Pour plus d'informations sur les autorisations supplémentaires pour Amazon ECR, consultez la section Autorisations Amazon ECR.

La tâche ne peut pas extraire '***nom de l'image***'depuis le référentiel Amazon ECR'***URI du référentiel***'. Il existe un problème de connexion entre la tâche et Amazon ECR. Vérifiez la configuration réseau de votre tâche.

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR. Vérifiez la connexion au *repository URI* référentiel.

Pour plus d'informations sur la manière de vérifier et de résoudre le problème, consultez la section Vérification de la connectivité de la tâche Amazon ECS arrêtée.

La tâche ne peut pas extraire l'image. Vérification de la configuration de votre réseau

Cette erreur indique que la tâche ne peut pas se connecter à Amazon ECR.

Pour plus d'informations sur la manière de vérifier et de résoudre le problème, consultez la section Vérification de la connectivité de la tâche Amazon ECS arrêtée.

CannotPullContainerError: le manifeste pull image a été réessayé 5 fois : échec de la résolution de la référence

Cette erreur indique que la tâche ne peut pas extraire l'image.

Pour résoudre cela, vous pouvez :

- Vérifiez que l'image spécifiée dans la définition de tâche correspond à l'image du référentiel.

- Amazon ECS impose la stabilité de la version de l'image. Si l'image d'origine n'est plus disponible, vous recevez ce message d'erreur. La balise d'image permet de renforcer ce comportement. Modifiez l'image dans la définition de la tâche en remplaçant la balise « :latest » par une version spécifique. Pour plus d'informations, consultez Résolution des images de conteneurs.

Pour plus d'informations sur la manière de vérifier et de résoudre le problème, consultez la section Vérification de la connectivité de la tâche Amazon ECS arrêtée.

Erreur d'API (500) : Obtenir https://111122223333.dkr.ecr.us-east-1.amazonaws.com/v2/: net/http: demande annulée en attente de connexion

Cette erreur indique qu'une connexion a expiré, car il n'existe pas d'itinéraire vers Internet.

Pour résoudre ce problème, vous pouvez :

- Pour les tâches dans les sous-réseaux publics, spécifiez ACTIVÉ pour Auto-assign IP publique lors du lancement de la tâche. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.
- Pour les tâches dans des sous-réseaux privés, spécifiez HANDICAPÉ pour Auto-assign IP publique lors du lancement de la tâche, et configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.

Erreur d'API

Cette erreur indique l'existence d'un problème de connexion avec le point de terminaison Amazon ECR.

Pour plus d'informations sur la façon de résoudre ce problème, consultez Comment puis-je résoudre l'erreur Amazon ECR ? »CannotPullContainerError: « erreur d'API » dans Amazon ECS sur le Support site web.

écrire/var/lib/docker/tmp/***GetImageBlob111111111***: il ne reste plus d'espace sur l'appareil

Cette erreur indique que l'espace disque est insuffisant.

Pour résoudre ce problème, vous devez libérer de l'espace disque.

Si vous utilisez Amazon ECS-optimized AMI, vous pouvez utiliser la commande suivante pour récupérer les 20 fichiers les plus volumineux de votre système de fichiers :

```
du -Sh / | sort -rh | head -20
```

Exemple de sortie :

```
5.7G    /var/lib/docker/
containers/50501b5f4cbf90b406e0ca60bf4e6d4ec8f773a6c1d2b451ed8e0195418ad0d2
1.2G    /var/log/ecs
594M    /var/lib/docker/devicemapper/mnt/
c8e3010e36ce4c089bf286a623699f5233097ca126ebd5a700af023a5127633d/rootfs/data/logs
...
```

Dans certains cas, le volume racine peut être saturé par un conteneur en cours d'exécution. Si le conteneur utilise le pilote de journal `json-file` par défaut sans limite `max-size`, le fichier journal est peut-être responsable de la plupart de l'utilisation de cet espace. Vous pouvez utiliser le `docker ps` commande pour vérifier quel conteneur utilise l'espace en mappant le nom du répertoire de la sortie à l'ID du conteneur. Par exemple :

```
CONTAINER ID   IMAGE                                COMMAND             CREATED
 STATUS              PORTS                               NAMES
50501b5f4cbf   amazon/amazon-ecs-agent:latest   "/agent"            4 days ago
 Up 4 days                                           ecs-agent
```

Par défaut, lorsque vous utilisez le pilote de journal `json-file`, Docker capture la sortie standard (et l'erreur standard) de tous vos conteneurs et les écrit dans les fichiers au format JSON. Vous pouvez définir l'option `max-size` comme une option de pilote de journal, ce qui empêche le fichier journal d'occuper trop d'espace. Pour plus d'informations, consultez la section Pilote de journalisation dans la documentation Docker.

Voici un extrait de définition de conteneur illustrant la manière d'utiliser cette option :

```
{
    "log-driver": "json-file",
    "log-opts": {
        "max-size": "256m"
    }
}
```

Si vos journaux de conteneur occupent trop d'espace disque, vous pouvez également utiliser le pilote de journal `awslogs`. Le `awslogs` log driver envoie les journaux à CloudWatch, ce qui permet de

libérer de l'espace disque qui serait autrement utilisé pour vos journaux de conteneurs sur l'instance de conteneur. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à CloudWatch.

Vous devrez peut-être mettre à jour la taille du disque auquel Docker peut accéder.

Pour plus d'informations, voir CannotPullContainerError: il ne reste plus d'espace sur l'appareil.

ERREUR : toomanyrequests : trop de requêtes ou vous avez atteint votre limite de taux d'extraction.

Cette erreur indique l'existence d'une limite de débit Docker Hub.

Si vous recevez un message pour l'une des erreurs suivantes, vous êtes probablement en train d'atteindre les limites de débit Docker Hub :

Pour en savoir plus sur les limites de débit Docker Hub, consultez Understanding Docker Hub rate limiting.

Si vous avez augmenté la limite de débit Docker Hub et que vous devez authentifier vos extractions Docker pour vos instances de conteneur, consultez la section Authentification de registre privé pour les instances de conteneur.

Réponse d'erreur du démon : Get ***url***: net/http: demande annulée en attente de connexion

Cette erreur indique qu'une connexion a expiré, car il n'existe pas d'itinéraire vers Internet.

Pour résoudre ce problème, vous pouvez :

- Pour les tâches dans les sous-réseaux publics, spécifiez ACTIVÉ pour Auto-assign IP publique lors du lancement de la tâche. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.
- Pour les tâches dans des sous-réseaux privés, spécifiez HANDICAPÉ pour Auto-assign IP publique lors du lancement de la tâche, et configurez une passerelle NAT dans votre VPC pour acheminer les demandes vers Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.

ref pull a été réessayé 1 fois : échec de la copie : httpReaderSeeker: échec de l'ouverture : code d'état inattendu

Cette erreur indique un échec lors de la copie d'une image.

Pour résoudre ce problème, consultez l'un des articles suivants :

- Pour les tâches Fargate, consultez Comment puis-je corriger l'erreur « cannotpullcontainererror » pour mes tâches Amazon ECS sur Fargate ?.
- Pour les autres tâches, consultez Comment puis-je corriger l'erreur « cannotpullcontainererror » pour mes tâches Amazon ECS ?.

accès en extraction refusé

Cette erreur indique l'impossibilité d'accéder à l'image.

Pour résoudre ce problème, vous devrez peut-être authentifier votre client Docker auprès d'Amazon ECR. Pour plus d'informations, consultez la section Authentification de registre privé dans le Guide de l'utilisateur Amazon ECR.

échec de la commande d'extraction : panique : erreur d'exécution : adresse mémoire non valide ou déréférencement de pointeur nul

Cette erreur indique qu'il n'est pas possible d'accéder à l'image en raison d'une adresse mémoire non valide ou d'un déréférencement de pointeur nul.

Pour résoudre ce problème :

- Vérifiez que vous disposez des règles du groupe de sécurité pour accéder à Amazon S3.
- Lorsque vous utilisez des points de terminaison de passerelle, vous devez ajouter une route dans la table de routage pour accéder au point de terminaison.

erreur d'extraction de l'image conf/error extraction de la configuration de l'image

Cette erreur indique qu'une limite de débit a été atteinte ou qu'il y a une erreur réseau :

Pour résoudre ce problème, consultez Comment puis-je résoudre le problème »CannotPullContainerError« erreur dans ma tâche de type de lancement Amazon ECS EC2.

Contexte annulé

Cette erreur indique que le contexte a été annulé.

La cause courante de cette erreur est que le VPC utilisé par votre tâche n'a pas de route pour extraire l'image d'Amazon ECR.

Vérification de la connectivité de la tâche Amazon ECS arrêtée

Il arrive qu'une tâche s'arrête en raison d'un problème de connectivité réseau. Il peut s'agir d'un problème intermittent, mais il est probablement dû au fait que la tâche ne peut pas se connecter à un point de terminaison.

Test de la connectivité de la tâche

Vous pouvez utiliser le dossier d'exploitation `AWSSupport-TroubleshootECSTaskFailedToStart` pour tester la connectivité de la tâche. Lorsque vous utilisez le dossier d'exploitation, vous avez besoin des informations suivantes sur les ressources :

- ID de la tâche

  Utilisez l'ID de la dernière tâche ayant échoué.
- Le cluster dans lequel se trouvait la tâche

Pour plus d'informations sur l'utilisation du dossier d'exploitation, consultez la section `AWSSupport-TroubleshootECSTaskFailedToStart` de la Référence du dossier d'exploitation d'AWS Systems Manager Automation.

Le dossier d'exploitation analyse la tâche. Vous pouvez consulter les résultats dans la section Sortie pour les problèmes suivants susceptibles d'empêcher le démarrage d'une tâche :

- Connectivité réseau avec le registre de conteneurs configuré
- Connectivité au point de terminaison de VPC
- Configuration des règles de groupe de sécurité

Résolution des problèmes liés aux points de terminaison du VPC

Lorsque le résultat du dossier d'exploitation `AWSSupport-TroubleshootECSTaskFailedToStart` indique le problème du point de terminaison de VPC, vérifiez la configuration suivante :

- Le VPC sur lequel vous créez le point de terminaison et le point de terminaison du VPC doivent utiliser un DNS privé.

- Assurez-vous que vous disposez d'un AWS PrivateLink point de terminaison du service auquel la tâche ne peut pas se connecter dans le même VPC que la tâche. Pour plus d'informations, consultez l'un des points suivants :

Service	Informations sur le point de terminaison VPC pour le service
Amazon ECR	Points de terminaison du VPC de l'interface Amazon ECR (AWS PrivateLink)
Systems Manager	Renforcement de la sécurité des instances EC2 à l'aide des points de terminaison de VPC pour Systems Manager
Secrets Manager	À l'aide d'un AWS Secrets Manager Point de terminaison VPC
CloudWatch	CloudWatch Point de terminaison d'un VPC
Amazon S3	AWS PrivateLink pour Amazon S3

- Configurez une règle de sortie pour le sous-réseau de tâches qui autorise le protocole HTTPS sur le trafic DNS (TCP) du port 443. Pour plus d'informations, consultez la section Configurer les règles de groupe de sécurité dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.
- Si vous utilisez un serveur de noms de domaine personnalisé, confirmez les paramètres de la requête DNS. La requête doit avoir un accès sortant sur le port 53 et utiliser les protocoles UDP et TCP. Il doit également disposer d'un accès HTTPS sur le port 443. Pour plus d'informations, consultez la section Configurer les règles de groupe de sécurité dans le Guide de l'utilisateur Amazon Elastic Compute Cloud.
- Si le sous-réseau possède une ACL réseau, les règles ACL suivantes sont requises :
  - Une règle sortante qui autorise le trafic sur les ports 1 024 à 65 535.

- Une règle entrante qui autorise le trafic TCP sur le port 443.

Pour plus d'informations sur la configuration des règles, consultez la section Contrôle du trafic vers les sous-réseaux à l'aide des ACL réseau dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Résolution des problèmes réseau

Lorsque le résultat du dossier d'exploitation `AWSSupport-TroubleshootECSTaskFailedToStart` indique le problème réseau, vérifiez la configuration suivante :

Tâches utilisant le mode réseau awsvpc dans un sous-réseau public

Effectuez la configuration suivante en fonction du dossier d'exploitation :

- Pour les tâches dans les sous-réseaux publics, spécifiez ACTIVÉ pour Auto-assign IP publique lors du lancement de la tâche. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.
- Vous avez besoin d'une passerelle pour gérer le trafic Internet. La table de routage du sous-réseau de tâches doit contenir une route pour le trafic vers la passerelle.

  Pour en savoir plus, consultez la section Ajout et suppression d'itinéraires d'une table de routage du Guide de l'utilisateur Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0,0,0.0/0	ID de passerelle NAT
Passerelle Internet	0,0,0.0/0	ID de la passerelle Internet

- Si le sous-réseau de la tâche possède une ACL réseau, les règles ACL suivantes sont requises :
  - Une règle sortante qui autorise le trafic sur les ports 1 024 à 65 535.
  - Une règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôle du trafic vers les sous-réseaux à l'aide des ACL réseau dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Tâches utilisant le mode réseau awsvpc dans un sous-réseau privé

Effectuez la configuration suivante en fonction du dossier d'exploitation :

- Choisissez HANDICAPÉ pour Auto-assign IP publique lors du lancement de la tâche.
- Configurez une passerelle NAT dans votre VPC pour acheminer les requêtes vers Internet. Pour plus d'informations, consultez la section Passerelles NAT dans le Guide de l'utilisateur Amazon Virtual Private Cloud.
- La table de routage du sous-réseau de tâches doit contenir une route pour le trafic vers la passerelle NAT.

  Pour en savoir plus, consultez la section Ajout et suppression d'itinéraires d'une table de routage du Guide de l'utilisateur Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0,0,0.0/0	ID de passerelle NAT

- Si le sous-réseau de la tâche possède une ACL réseau, les règles ACL suivantes sont requises :
  - Une règle sortante qui autorise le trafic sur les ports 1 024 à 65 535.
  - Une règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôle du trafic vers les sous-réseaux à l'aide des ACL réseau dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Tâches n'utilisant pas le mode réseau awsvpc dans un sous-réseau public

Effectuez la configuration suivante en fonction du dossier d'exploitation :

- Choisissez Activer pour Attribuer automatiquement l'adresse IP sous Mise en réseau pour les instances Amazon EC2 lorsque vous créez le cluster.

  Cette option attribue une adresse IP publique à l'interface réseau principale de l'instance.
- Vous avez besoin d'une passerelle pour gérer le trafic Internet. La table de routage du sous-réseau de l'instance doit contenir une route pour le trafic vers la passerelle.

Pour en savoir plus, consultez la section Ajout et suppression d'itinéraires d'une table de routage du Guide de l'utilisateur Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0,0,0.0/0	ID de passerelle NAT
Passerelle Internet	0,0,0.0/0	ID de la passerelle Internet

- Si le sous-réseau de l'instance possède une ACL réseau, les règles ACL suivantes sont requises :
  - Une règle sortante qui autorise le trafic sur les ports 1 024 à 65 535.
  - Une règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôle du trafic vers les sous-réseaux à l'aide des ACL réseau dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Tâches n'utilisant pas le mode réseau awsvpc dans un sous-réseau privé

Effectuez la configuration suivante en fonction du dossier d'exploitation :

- Choisissez Désactiver pour Attribuer automatiquement l'adresse IP sous Mise en réseau pour les instances Amazon EC2 lorsque vous créez le cluster.
- Configurez une passerelle NAT dans votre VPC pour acheminer les requêtes vers Internet. Pour plus d'informations, veuillez consulter NAT Gateways (Passerelles NAT) dans le Guide de l'utilisateur Amazon VPC.
- La table de routage du sous-réseau de l'instance doit contenir une route pour le trafic vers la passerelle NAT.

  Pour en savoir plus, consultez la section Ajout et suppression d'itinéraires d'une table de routage du Guide de l'utilisateur Amazon Virtual Private Cloud.

Type de passerelle	Destination de la table de routage	Cible de la table de routage
NAT	0,0,0.0/0	ID de passerelle NAT

- Si le sous-réseau de la tâche possède une ACL réseau, les règles ACL suivantes sont requises :
  - Une règle sortante qui autorise le trafic sur les ports 1 024 à 65 535.
  - Une règle entrante qui autorise le trafic TCP sur le port 443.

  Pour plus d'informations sur la configuration des règles, consultez la section Contrôle du trafic vers les sous-réseaux à l'aide des ACL réseau dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Affichage des requêtes de rôle IAM pour des tâches Amazon ECS

Lorsque vous utilisez un fournisseur pour vos informations d'identification de tâche dans un rôle IAM, les requêtes du fournisseur sont enregistrées dans un journal d'audit. Le journal d'audit hérite des mêmes paramètres de rotation du journal que le journal de l'agent de conteneur. Les variables `ECS_LOG_ROLLOVER_TYPE`, `ECS_LOG_MAX_FILE_SIZE_MB` et `ECS_LOG_MAX_ROLL_COUNT` de configuration de l'agent de conteneur peuvent être définies de façon à affecter le comportement du journal d'audit. Pour plus d'informations, consultez Paramètres de configuration du journal de l'agent de conteneur Amazon ECS.

Pour l'agent de conteneur version 1.36.0 et ultérieure, le journal d'audit se trouve à l'adresse `/var/log/ecs/audit.log`. Lorsque le journal est tourné, un horodatage au format *`YYYY`*`-`*`MM`*`-`*`DD`*`-`*`HH`* est ajouté à la fin du nom du fichier journal.

Pour l'agent de conteneur version 1.35.0 et antérieure, le journal d'audit se trouve à l'adresse `/var/log/ecs/audit.log.`*`YYYY`*`-`*`MM`*`-`*`DD`*`-`*`HH`*.

Le format d'entrée de journal est le suivant :

- Horodatage
- Code de réponse HTTP
- Numéro de port et adresse IP d'origine de la demande
- URI relative du fournisseur d'informations d'identification

- L'agent utilisateur qui a effectué la demande
- L'ARN de la tâche à laquelle appartient le conteneur demandeur
- Le nom d'API `GetCredentials` et son numéro de version
- Le nom du cluster Amazon ECS dans lequel l'instance de conteneur est enregistrée
- L'ARN d'instance de conteneur

Vous pouvez utiliser la commande suivante pour visualiser les fichiers journaux.

```
cat /var/log/ecs/audit.log.2016-07-13-16
```

Sortie :

```
2016-07-13T16:11:53Z 200 172.17.0.5:52444 "/v1/credentials" "python-requests/2.7.0
 CPython/2.7.6 Linux/4.4.14-24.50.amzn1.x86_64" TASK_ARN GetCredentials
 1 CLUSTER_NAME CONTAINER_INSTANCE_ARN
```

Affichage des messages d'événement d'un service Amazon ECS

Lorsque vous voulez résoudre un problème lié à un service, le premier endroit à consulter pour obtenir des informations est le journal des événements du service. Vous pouvez consulter les événements de service à l'aide du `DescribeServices` API, le AWS CLI, ou en utilisant le Console de gestion AWS.

Lorsque vous affichez des messages d'événements de service à l'aide de l'API Amazon ECS, seuls les événements du planificateur de service sont renvoyés. Ceux-ci comprennent le placement des tâches les plus récents et les événements d'état de l'instance. Toutefois, la console Amazon ECS affiche les événements de service à partir des sources suivantes.

- Placement des tâches et événements d'état de l'instance à partir du planificateur de service Amazon ECS service. Ces événements ont un préfixe de service ***(service-name)***. Pour que cette vue des événements soit utile, seuls les `100` événements les plus récents sont affichés. Les messages d'événement en double sont omis jusqu'à ce que la cause soit résolue ou que six heures se soient écoulées. Si la cause n'est pas résolue dans les six heures, vous recevez un autre message d'événement de service.

- Événements de service Auto Scaling. Ces événements ont pour préfixe Message et ne se produisent que lorsqu'un service est configuré avec une stratégie de mise à l'échelle automatique des applications.

> **Tip**
>
> Vous pouvez utiliser le Serveur Amazon ECS MCP avec des assistants IA pour analyser les événements de service à l'aide du langage naturel.

Procédez comme suit pour afficher vos messages d'événement de service actuels.

Console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sur la page Clusters, choisissez le cluster.
4. Choisissez le service à inspecter.
5. Sur le Évènements onglet, affiche les messages.

AWS CLI

Utilisez la commande describe-services pour afficher les messages d'événement de service pour un service spécifié.

Les suivants AWS CLI l'exemple décrit *service-name* service dans le *default* cluster, qui fournira les derniers messages relatifs aux événements de service.

```
aws ecs describe-services \
    --cluster default \
    --services service-name \
    --region us-west-2
```

Messages d'événement de service Amazon ECS

Voici quelques exemples de messages d'événement de service que vous pouvez rencontrer dans la console Amazon ECS.

service (*service-name*) a atteint un état stable.

Le planificateur de service envoie un événement de service `service (service-name) has reached a steady state.` lorsque le service est sain et au nombre de tâches souhaité, atteignant ainsi un état stable.

Le planificateur de service rapporte l'état de façon régulière, vous pouvez donc recevoir ce message plusieurs fois.

service (*service-name*) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences.

Le planificateur de service envoie ce message d'événement lorsqu'il n'a pas trouvé les ressources disponibles pour ajouter une autre tâche. Les causes possibles sont les suivantes :

Utilisez des fournisseurs de capacité pour mettre automatiquement à l'échelle vos instances EC2. Pour plus d'informations, consultez Fournisseurs de capacité Amazon ECS pour les charges de travail EC2.

> Si vous aviez l'intention de faire appel à un fournisseur de capacité, assurez-vous que vous adoptez une stratégie de fournisseur de capacité ou que vous avez une stratégie de fournisseur de capacité par défaut associée à votre cluster, et que vous ne transmettez pas le type de lancement et la stratégie du fournisseur de capacité en entrée

Aucune instance de conteneur n'a été trouvée dans votre cluster

> Si aucune instance de conteneur n'est enregistrée dans le cluster dans lequel vous tentez d'exécuter une tâche, vous recevez cette erreur. Vous devez ajouter des instances de conteneur à votre cluster. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Nombre de ports insuffisant

> Si votre tâche utilise le mappage de port hôte fixe (si, par exemple, elle emploie le port 80 sur l'hôte pour un serveur web), vous devez avoir au moins une instance de conteneur par tâche, car un seul conteneur peut utiliser un même port hôte à la fois. Vous devez ajouter des instances de conteneur à votre cluster ou réduire votre nombre de tâches souhaitées.

Nombre de ports enregistrés trop important

L’instance de conteneur correspondante la plus proche pour le placement des tâches ne peut pas dépasser la limite maximale autorisée de ports réservés, soit 100 ports hôtes par instance de conteneur. L'utilisation du mappage de port hôte dynamique peut résoudre le problème.

Port déjà utilisé

La définition de cette tâche utilise le même port dans son mappage de ports qu’une tâche en cours d’exécution sur l’instance de conteneur qui a été choisie. Le message d'événement de service aurait l'ID d'instance de conteneur choisi dans le message suivant.

```
The closest matching container-instance is already using a port required by your
 task.
```

Mémoire insuffisante

Si votre définition de tâche spécifie 1 000 Mio de mémoire et que les instances de conteneur de votre cluster ont chacune 1 024 Mio de mémoire, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins de mémoire dans votre définition de tâche afin de pouvoir lancer plusieurs tâches par instance de conteneur, ou lancer plusieurs instances de conteneur dans votre cluster.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

UC insuffisante

Une instance de conteneur comporte 1 024 unités d'UC pour chaque cœur de processeur. Si votre définition de tâche spécifie 1 000 unités d'UC, et que les instances de conteneur de votre cluster ont chacune 1 024 unités d'UC, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins d'unités d'UC dans votre définition de tâche pour pouvoir lancer plusieurs tâches par instance de conteneur ou plusieurs instances de conteneur dans votre cluster.

Pas suffisamment de points d'attache ENI disponibles

Les tâches qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'ENI qui peuvent être attachées aux instances Amazon EC2 est limité. Il n'y a pas d'instances de conteneur dans le cluster qui présente une capacité ENI disponible.

La limite ENI pour les instances de conteneur individuelles varie selon les conditions suivantes :

- Si vous n'avez pas opté pour le paramètre de compte `awsvpcTrunking`, la limite ENI pour chaque instance de conteneur dépend du type de l'instance. Pour de plus amples informations, consultez Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.
- Si vous avez opté pour le paramètre de compte `awsvpcTrunking`, mais que vous n’avez pas lancé de nouvelles instances de conteneur utilisant un type d’instance pris en charge après ce choix, la limite ENI pour chaque instance de conteneur reste la valeur par défaut. Pour de plus amples informations, consultez Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.
- Si vous avez opté pour le paramètre de compte `awsvpcTrunking` et que vous avez lancé de nouvelles instances de conteneur utilisant un type d'instance pris en charge après ce choix, des ENI supplémentaires sont disponibles. Pour plus d’informations, consultez Instances prises en charge pour augmenter le nombre d’interfaces réseau de conteneurs Amazon ECS.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Augmentation des interfaces réseau d’une instance de conteneur Amazon ECS Linux.

Vous pouvez ajouter des instances de conteneur à votre cluster afin de mettre à disposition davantage de cartes réseau.

Attribut requis manquant dans l'instance de conteneur

Certains paramètres de définition de tâche nécessitent l'installation d'une version spécifique de l'API Docker à distance sur l'instance de conteneur. D'autres, telles que les options du pilote de journalisation, exigent que les instances de conteneur enregistrent ces pilotes de journal avec la variable de configuration d'agent `ECS_AVAILABLE_LOGGING_DRIVERS`. Si votre définition de tâche contient un paramètre qui nécessite un attribut d’instance de conteneur spécifique, et que vous ne disposez pas d’instances de conteneur répondant à cette exigence, la tâche en question ne peut pas être placée.

Cette erreur est souvent due au fait que votre service utilise des tâches utilisant le mode réseau `awsvpc` et EC2. Aucune instance de conteneur n'est enregistrée sur le cluster dans le même sous-réseau que celui spécifié dans `awsvpcConfiguration` lors de la création du service.

Vous pouvez utiliser le AWSSupport-TroubleshootECSContainerInstance runbook pour résoudre les problèmes. Le dossier d'exploitation vérifie si les données utilisateur de l'instance contiennent les informations de cluster correctes, si le profil d'instance contient les autorisations requises et vérifie les problèmes de configuration réseau. Pour plus d'informations, consultez AWSSupport-TroubleshootECSContainerInstance dans le AWS Systems Manager Guide de l'utilisateur de la référence du dossier d'exploitation d'automatisation.

Pour plus d'informations sur les attributs nécessaires pour les paramètres de définition de tâche spécifiques et les variables de configuration d'agent, consultez Paramètres de définition de tâche Amazon ECS pour Fargate et Configuration de l'agent de conteneur Amazon ECS.

service (***service-name***) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. L'instance de conteneur correspondante la plus proche ***ID d'instance du conteneur*** n'a pas suffisamment de processeurs disponibles.

L'instance de conteneur qui correspond le mieux au placement de la tâche ne contient pas suffisamment d'unités d'UC pour répondre aux exigences de la définition de la tâche. Vérifiez les exigences en matière d'UC dans les paramètres de définition de conteneur et de taille de tâche de la définition de tâche.

service (***service-name***) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences. L'instance de conteneur correspondante la plus proche ***ID d'instance du conteneur*** erreur « AGENT » rencontrée.

L'agent de conteneur Amazon ECS de l'instance de conteneur correspondante la plus proche pour le placement des tâches est déconnecté. Si vous pouvez vous connecter à l'instance de conteneur avec SSH, vous pouvez examiner les journaux d'agent. Pour plus d'informations, consultez Paramètres de configuration du journal de l'agent de conteneur Amazon ECS. Vous devez également vérifier que l'agent est en cours d'exécution sur l'instance. Si vous utilisez Amazon ECS-optimized AMI, vous pouvez essayer d'arrêter et de redémarrer l'agent à l'aide de la commande suivante.

- Pour l'Amazonie ECS-optimized Amazon Linux 2 AMI et Amazon ECS-optimized AMI Amazon Linux 2023

```
sudo systemctl restart ecs
```

- Pour l'Amazonie ECS-optimized AMI Amazon Linux

```
sudo stop ecs && sudo start ecs
```

service (*service-name*) (tâche *identifiant de tâche*) (exemple *instance-id*) est mauvais pour la santé (elb) *nom elb*) pour (raison pour laquelle l'instance a échoué au moins UnhealthyThreshold nombre de bilans de santé consécutifs.)

Ce service est enregistré avec un équilibreur de charge et les surveillances de l'état de cet équilibreur échouent. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les surveillances de l'état échouent. Pour plus d'informations, consultez Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS.

service (*service-name*) n'est pas en mesure de démarrer les tâches correctement de manière cohérente.

Ce service contient des tâches qui n'ont pas pu démarrer après plusieurs tentatives consécutives. À ce stade, le planificateur de service commence à augmenter progressivement le délai entre les tentatives. Vous devez déterminer pourquoi le lancement de vos tâches échoue. Pour plus d'informations, consultez Logique de limitation d'un service Amazon ECS.

Une fois le service mis à jour (mise à jour de la définition de tâche, par exemple), le planificateur de service se comporte de nouveau normalement.

service (*service-name*) les opérations sont limitées. Réessayez ultérieurement.

Ce service ne peut pas lancer plus de tâches à cause des limitations de l'API. Lorsque le planificateur de service peut lancer plus de tâches, il reprend.

Pour demander une augmentation de quota de limite de débit d'API, ouvrez la page du Centre AWS Support, connectez-vous si nécessaire et choisissez Create case (Créer une demande). Sélectionnez Service Limit increase (Augmentation des limites de service). Remplissez et envoyez le formulaire.

service (***service-name***) n'a pas pu arrêter ou démarrer des tâches pendant un déploiement en raison de la configuration de déploiement du service. Mettre à jour le minimumHealthyPercent ou la valeur MaximumPercent et réessayez.

Ce service ne peut pas arrêter ou démarrer des tâches pendant un déploiement de service en raison de la configuration du déploiement. La configuration de déploiement se compose des valeurs `minimumHealthyPercent` et `maximumPercent` qui sont définies lorsque le service est créé. Ces valeurs peuvent également être mises à jour sur un service existant.

`minimumHealthyPercent` représente la limite inférieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée. Il s'agit d'un pourcentage du nombre de tâches souhaité pour le service. Cette valeur est arrondie à la valeur supérieure. Par exemple, si le pourcentage minimum sain est `50` et que le nombre de tâches souhaité est de quatre, le planificateur peut arrêter deux tâches existantes avant de démarrer deux nouvelles tâches. De même, si le pourcentage de santé minimum est de 75 % et que le nombre de tâches souhaité est de deux, le planificateur ne peut pas arrêter de tâche car la valeur résultante est également de deux.

Le `maximumPercent` représente la limite inférieure du nombre de tâches qui doivent être exécutées pour un service pendant un déploiement ou lorsqu'une instance de conteneur est drainée. Il s'agit d'un pourcentage du nombre de tâches souhaité pour un service. Cette valeur est arrondie à la valeur inférieure. Par exemple, si le pourcentage maximal est `200` et que le nombre de tâches souhaitées est quatre, le planificateur peut démarrer quatre nouvelles tâches avant d'arrêter quatre tâches existantes. De même, si le pourcentage maximal est `125` et que le nombre de tâches souhaité est de trois, le planificateur ne peut pas démarrer de tâche car la valeur résultante est également de trois.

Lorsque vous définissez un pourcentage d'état minimum ou un pourcentage maximal, vous devez vous assurer que le planificateur peut arrêter ou démarrer au moins une tâche lorsqu'un déploiement est déclenché.

service (***service-name***) n'a pas pu placer de tâche. Motif : Vous avez atteint la limite du nombre de tâches que vous pouvez exécuter simultanément

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour plus d'informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (***service-name***) n'a pas pu placer de tâche. Motif : Erreur interne.

Voici la raison possible de cette erreur :

Le service ne peut pas démarrer une tâche car un sous-réseau se trouve dans une zone de disponibilité non prise en charge.

Pour plus d'informations sur les Régions Fargate et les zones de disponibilités prises en charge, consultez the section called “AWS Régions de Fargate”.

Pour plus d'informations sur la façon d'afficher la zone de disponibilité de sous-réseau, consultez Afficher votre sous-réseau dans le Guide de l'utilisateur Amazon VPC.

service (***service-name***) n'a pas pu placer de tâche. Motif : la configuration CPU demandée est supérieure à votre limite.

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour plus d’informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (***service-name***) n'a pas pu placer de tâche. Motif : la configuration MEMORY demandée est supérieure à votre limite.

Vous pouvez demander une augmentation de quota pour la ressource qui a provoqué l'erreur. Pour plus d’informations, consultez Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (***service-name***) n'a pas pu placer de tâche. Motif : Vous avez atteint la limite du nombre des vCPU que vous pouvez exécuter simultanément

AWS Fargate est en train de passer des quotas basés sur le nombre de tâches à vCPU-based quotas.

Vous pouvez demander une augmentation de quota pour Fargate v.CPU-based quota. Pour plus d’informations, consultez Service Quotas. Pour demander une augmentation de quota Fargate, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

service (*service-name*) n'a pas pu atteindre l'état stable car l'ensemble de tâches (*tâcheSet-ID*) n'a pas pu s'adapter. Raison : le nombre de tâches protégées est supérieur au nombre souhaité de tâches.

Le service a plus de tâches protégées que le nombre souhaité de tâches. Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la protection des tâches en cours expire pour pouvoir y mettre fin.
- Déterminez les tâches qui peuvent être arrêtées et utilisez l'API `UpdateTaskProtection` avec l'option `protectionEnabled` définie sur `false` pour désactiver la protection de ces tâches.
- Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

service (*service-name*) n'a pas pu atteindre l'état d'équilibre. Motif : aucune instance de conteneur n'a été trouvée dans votre fournisseur de capacité.

Le planificateur de service envoie ce message d'événement lorsqu'il n'a pas trouvé les ressources disponibles pour ajouter une autre tâche. Les causes possibles sont les suivantes :

Aucun fournisseur de capacité n'est associé au cluster

Utilisez `describe-services` pour vérifier qu'un fournisseur de capacité est associé au cluster. Vous pouvez mettre à jour la stratégie du fournisseur de capacité pour le service.

Vérifiez que le fournisseur de capacité dispose de capacité. Dans le cas d'EC2, assurez-vous que les instances de conteneur répondent aux exigences de définition des tâches.

Aucune instance de conteneur n'a été trouvée dans votre cluster

Si aucune instance de conteneur n'est enregistrée dans le cluster dans lequel vous tentez d'exécuter une tâche, vous recevez cette erreur. Vous devez ajouter des instances de conteneur à votre cluster. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.

Nombre de ports insuffisant

Si votre tâche utilise le mappage de port hôte fixe (si, par exemple, elle emploie le port 80 sur l'hôte pour un serveur Web), vous devez avoir au moins une instance de conteneur par tâche. Un seul conteneur peut utiliser un même port hôte à la fois. Vous devez ajouter des instances de conteneur à votre cluster ou réduire votre nombre de tâches souhaitées.

Nombre de ports enregistrés trop important

L'instance de conteneur correspondante la plus proche pour le placement des tâches ne peut pas dépasser la limite maximale autorisée de ports réservés, soit 100 ports hôtes par instance de conteneur. L'utilisation du mappage de port hôte dynamique peut résoudre le problème.

Port déjà utilisé

La définition de cette tâche utilise le même port dans son mappage de ports qu'une tâche en cours d'exécution sur l'instance de conteneur qui a été choisie. Le message d'événement de service aurait l'ID d'instance de conteneur choisi dans le message suivant.

```
The closest matching container-instance is already using a port required by your
 task.
```

Mémoire insuffisante

Si votre définition de tâche spécifie 1 000 Mio de mémoire et que les instances de conteneur de votre cluster ont chacune 1 024 Mio de mémoire, vous ne pouvez exécuter qu'une seule copie de cette tâche par instance de conteneur. Vous pouvez essayer avec moins de mémoire dans votre définition de tâche afin de pouvoir lancer plusieurs tâches par instance de conteneur, ou lancer plusieurs instances de conteneur dans votre cluster.

> **Note**
>
> Si vous essayez d'optimiser l'utilisation de vos ressources en fournissant à vos tâches autant de mémoire que possible pour un type d'instance particulier, consultez Réservation de mémoire pour une instance de conteneur Amazon ECS Linux.

Pas suffisamment de points d'attache ENI disponibles

Les tâches qui utilisent le mode réseau `awsvpc` reçoivent chacune leur propre interface réseau Elastic (ENI), qui est attachée à l'instance de conteneur qui l'héberge. Le nombre d'ENI qui peuvent être attachées aux instances Amazon EC2 est limité. Il n'y a pas d'instances de conteneur dans le cluster qui présente une capacité ENI disponible.

La limite ENI pour les instances de conteneur individuelles varie selon les conditions suivantes :

- Si vous n'avez pas opté pour le paramètre de compte `awsvpcTrunking`, la limite ENI pour chaque instance de conteneur dépend du type de l'instance. Pour de plus amples informations,

consultez Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.

- Si vous avez opté pour le paramètre de compte `awsvpcTrunking`, mais que vous n'avez pas lancé de nouvelles instances de conteneur utilisant un type d'instance pris en charge après ce choix, la limite ENI pour chaque instance de conteneur reste la valeur par défaut. Pour de plus amples informations, consultez Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.
- Si vous avez opté pour le paramètre de compte `awsvpcTrunking` et que vous avez lancé de nouvelles instances de conteneur utilisant un type d'instance pris en charge après ce choix, des ENI supplémentaires sont disponibles. Pour plus d'informations, consultez Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.

Pour en savoir plus sur l'acceptation du paramètre de compte `awsvpcTrunking`, consultez Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux.

Vous pouvez ajouter des instances de conteneur à votre cluster afin de mettre à disposition davantage de cartes réseau.

Attribut requis manquant dans l'instance de conteneur

Certains paramètres de définition de tâche nécessitent l'installation d'une version spécifique de l'API Docker à distance sur l'instance de conteneur. D'autres, telles que les options du pilote de journalisation, exigent que les instances de conteneur enregistrent ces pilotes de journal avec la variable de configuration d'agent `ECS_AVAILABLE_LOGGING_DRIVERS`. Si votre définition de tâche contient un paramètre qui nécessite un attribut d'instance de conteneur spécifique, et que vous ne disposez pas d'instances de conteneur répondant à cette exigence, la tâche en question ne peut pas être placée.

Cette erreur survient souvent si votre service a recours à des tâches qui utilisent le mode réseau `awsvpc` et EC2, et si aucune instance de conteneur n'est enregistrée sur le cluster dans le même sous-réseau que celui spécifié dans `awsvpcConfiguration` lors de la création du service.

Vous pouvez utiliser le AWSSupport-TroubleshootECSContainerInstance runbook pour résoudre les problèmes. Le dossier d'exploitation vérifie si les données utilisateur de l'instance contiennent les informations de cluster correctes, si le profil d'instance contient les autorisations requises et vérifie les problèmes de configuration réseau. Pour plus d'informations, consultez AWSSupport-TroubleshootECSContainerInstance dans le AWS Systems Manager Guide de l'utilisateur de la référence du dossier d'exploitation d'automatisation.

Pour plus d'informations sur les attributs nécessaires pour les paramètres de définition de tâche spécifiques et les variables de configuration d'agent, consultez Paramètres de définition de tâche Amazon ECS pour Fargate et Configuration de l'agent de conteneur Amazon ECS.

service (*service-name*) n'a pas pu placer de tâche. Motif : la capacité n'est pas disponible pour le moment. Veuillez réessayer ultérieurement ou dans une autre zone de disponibilité.

Il n'y a actuellement aucune capacité disponible pour exécuter votre service.

Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la capacité Fargate ou les instances de conteneur EC2 soient disponibles.
- Relancez le service et spécifiez des sous-réseaux supplémentaires.

service (*service-name*) échec du déploiement : les tâches n'ont pas pu démarrer.

Les tâches de votre service n’ont pas pu démarrer.

Pour plus d’informations sur le débogage des tâches arrêtées, consultez la section Messages d’erreurs liées aux tâches Amazon ECS arrêtées.

service (*service-name*) Le délai d'attente du démarrage de l'agent Amazon ECS a expiré. Veuillez consulter les journaux sur/var/log/ecs/ecs-agent.log ».

L'agent de conteneur Amazon ECS de l'instance de conteneur correspondante la plus proche pour le placement des tâches est déconnecté. Si vous pouvez vous connecter à l’instance de conteneur avec SSH, vous pouvez examiner les journaux d’agent. Pour plus d’informations, consultez Paramètres de configuration du journal de l’agent de conteneur Amazon ECS. Vous devez également vérifier que l'agent est en cours d'exécution sur l'instance. Si vous utilisez Amazon ECS-optimized AMI, vous pouvez essayer d'arrêter et de redémarrer l'agent à l'aide de la commande suivante.

- Pour l'Amazonie ECS-optimized AMI Amazon Linux 2

```
sudo systemctl restart ecs
```

- Pour l'Amazonie ECS-optimized AMI Amazon Linux

```
sudo stop ecs && sudo start ecs
```

service (***service-name***) ensemble de tâches (***tâcheSet-ID***) (tâche ***identifiant de tâche***) n'est pas en bonne santé dans le groupe cible (***cibleGroup-ARN)***) en raison de **LE GROUPE CIBLE N'EST PAS TROUVÉ**.

La tâche définie pour le service échoue aux surveillances de l'état, car le groupe cible est introuvable. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les surveillances de l'état échouent. Vous devez supprimer et recréer le service. Ne supprimez aucun groupe cible Elastic Load Balancing, à moins que le service Amazon ECS correspondant ne soit déjà supprimé.

service (***service-name***) ensemble de tâches (***tâcheSet-ID***) (tâche ***identifiant de tâche***) n'est pas en bonne santé dans le groupe cible (***cibleGroup-ARN)***) en raison de **LA CIBLE N'EST PAS TROUVÉE**.

La tâche définie pour le service échoue aux surveillances de l'état, car la cible est introuvable. Le message inclut l'ID de tâche pour aider à identifier la tâche spécifique pour laquelle les surveillances de l'état échouent.

Les politiques d'autorisation IAM ont été mal configurées ou modifiées, et ECS ne peut plus maintenir votre service

Le service n'est pas en mesure de gérer les tâches en raison de politiques d'autorisation IAM mal configurées ou modifiées. Le rôle IAM associé à votre service ou à vos tâches ECS ne dispose peut-être pas des autorisations requises.

Pour résoudre ce problème, ajoutez les autorisations nécessaires au rôle IAM. Pour plus d'informations sur la gestion des autorisations, consultez la section Ajout et suppression d'authorisations IAM dans le Guide de l'utilisateur IAM.

La relation de confiance IAM a été mal configurée ou modifiée, et ECS ne peut plus maintenir votre service.

Le service n'est pas en mesure de gérer les tâches en raison d'une relation de confiance IAM mal configurée ou modifiée. Le rôle IAM associé à votre service ou à vos tâches ECS peut avoir une stratégie d'approbation incorrecte.

Pour résoudre ce problème, configurez une stratégie d'approbation pour le rôle utilisé dans la définition de votre tâche. Pour plus d'informations sur la création de politiques de confiance pour les rôles personnalisés, consultez la section Création d'un rôle pour un cas d'utilisation personnalisé dans le Guide de l'utilisateur IAM.

service (*service-name*) n'a pas pu démarrer *number* tâches de déploiement *identifiant de déploiement*.

Le planificateur de services envoie ce message d'événement lorsqu'un flux de déploiement démarre correctement certaines tâches mais ne parvient pas à lancer toutes les tâches demandées en raison d'erreurs de capacité insuffisante. Cela se produit généralement lorsque le disjoncteur est activé et fournit une visibilité sur les raisons pour lesquelles les déploiements peuvent échouer ou revenir en arrière.

Le message inclut la raison spécifique de la panne, telle qu'une insuffisance du processeur, de la mémoire ou d'autres contraintes de ressources. Cela vous permet de comprendre quelles ressources doivent être utilisées pour résoudre le problème de déploiement.

Pour plus d'informations, consultez service (*service-name*) n'a pas pu placer de tâche car aucune instance de conteneur ne répondait à toutes ses exigences..

service (*service-name*) n'a pas pu placer de tâches dans votre cluster car la limite de capacité de provisionnement des tâches a été dépassée.

Le planificateur de services envoie ce message d'événement lorsque votre cluster a atteint la limite de 500 tâches pouvant figurer dans `PROVISIONING` état simultanément. Il s'agit d'une limite au niveau du cluster, et non d'un problème spécifique à un service.

Cela se produit généralement lorsque vous démarrez un service comportant un grand nombre de tâches souhaitées avec une capacité pré-provisionnée limitée, ou lorsque plusieurs services sont démarrés simultanément, ce qui entraîne un taux de rotation élevé des tâches.

Pour résoudre ce problème :

- Attendez que le provisionnement des tâches existantes soit terminé et passez au `RUNNING` état.
- Envisagez de faire évoluer vos services de manière plus progressive pour éviter d'atteindre la limite de provisionnement.
- Passez en revue la configuration du fournisseur de capacité de votre cluster pour vous assurer que des ressources adéquates sont disponibles.

Pour plus d'informations sur les quotas de service Amazon ECS, consultez Points de terminaison et quotas Amazon Elastic Container Service dans le Référence générale d'Amazon Web Services.

Messages d’événement relatifs à un dysfonctionnement d’un service Amazon ECS

Voici des exemples de messages d’événements de dysfonctionnement du service.

EC2 : (entretien) ***service-name***) (tâche ***identifiant de tâche***) (exemple ***instance-id***) (port ***numéro de port***) est mauvais pour la santé dans (groupe cible) ***nom-du-groupe-cible***) pour (raison) ***raison de l'échec***)

Ce message indique qu’une tâche exécutée sur une instance EC2 échoue aux surveillances de l’état. Pour plus d’informations, consultez les ressources suivantes :

- Comment puis-je faire en sorte que mes tâches Amazon ECS qui utilisent EC2 réussissent la surveillance de l’état de l’Application Load Balancer ?

EC2 avec TaskSet : (service) ***service-name***, Ensemble de tâches ***tâcheSet-id***) (tâche ***identifiant de tâche***) (exemple ***instance-id***) (port ***numéro de port***) est mauvais pour la santé dans (groupe cible) ***nom-du-groupe-cible***) pour (raison) ***raison de l'échec***)

Ce message indique qu’une tâche d’un jeu de tâches exécutée sur une instance EC2 échoue aux surveillances de l’état. Pour plus d’informations, consultez les ressources suivantes :

- Comment puis-je faire en sorte que mes tâches Amazon ECS qui utilisent EC2 réussissent la surveillance de l’état de l’Application Load Balancer ?

Tarif : (service) ***service-name***) (tâche ***identifiant de tâche***) (port ***numéro de port***) est mauvais pour la santé dans (groupe cible) ***nom-du-groupe-cible***) pour (raison) ***raison de l'échec***)

Ce message indique qu’une tâche Fargate échoue aux surveillances de l’état.

Pour plus d'informations sur la résolution des échecs de surveillance de l'état dans les tâches Fargate, consultez la section Comment résoudre les échecs de surveillance de l'état pour les tâches Amazon ECS sur Fargate ?.

Fargate avec TaskSet : (service ***service-name***, Ensemble de tâches ***tâcheSet-id***) (tâche ***identifiant de tâche***) (port ***numéro de port***) est mauvais pour la santé dans (groupe cible) ***nom-du-groupe-cible***) pour (raison) ***raison de l'échec***)

Ce message indique qu'une tâche d'un jeu de tâches exécuté sur Fargate échoue aux surveillances de l'état.

Pour plus d'informations sur la résolution des échecs de surveillance de l'état dans les tâches Fargate, consultez la section Comment résoudre les échecs de surveillance de l'état pour les tâches Amazon ECS sur Fargate ?.

Messages d'événement de rééquilibrage de service entre zones de disponibilité Amazon ECS

Voici quelques exemples de messages d'événement de service que vous pouvez rencontrer :

service (***service-name***) n'est pas équilibré en Z avec ***tâches numériques*** tâches dans ***Zone de disponibilité 1***, ***tâches numériques*** in ***Zone de disponibilité 2***, et ***tâches numériques*** in ***Zone de disponibilité 3***. AZ Rééquilibrage en cours.

Le planificateur de service envoie un événement de service `service (service-name) is not AZ balanced` lorsque le nombre de tâches n'est pas réparti uniformément entre les zones de disponibilité. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service (***service-name***) est équilibré en A avec ***tâches numériques*** tâches dans ***Zone de disponibilité 1***, ***tâches numériques*** tâches dans ***Zone de disponibilité 2***, et ***tâches numériques*** tâches dans ***Zone de disponibilité 3***.

Le planificateur de service envoie un événement de service `service (service-name) is AZ balanced` lorsque le rééquilibrage des services entre zones de disponibilité est terminé. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service-name a commencé *tâches numériques* tâches dans *Zone de disponibilité* à AZ Rebalance : *identifiants de tâches*.

Le planificateur de services envoie un *service-name*/*task-set-name* a commencé *number* tâches dans *Availability Zone* événement de service lorsqu'il démarre des tâches dans une zone de disponibilité en raison du rééquilibrage des services. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service-name s'est arrêté *tâches numériques* exécution de tâches dans *Zone de disponibilité* grâce au rééquilibrage AZ : *identifiant de tâche*.

Le planificateur de services envoie un *service-name*/*task-set-name* s'est arrêté *number* tâches dans *Availability Zone* événement de service lorsqu'il arrête des tâches dans une zone de disponibilité en raison d'un rééquilibrage des services. Il n'y a aucune mesure à prendre. Il s'agit d'un événement d'information.

service (*service-name*) ne parvient pas à placer une tâche dans *Zone de disponibilité* car aucune instance de conteneur ne répondait à toutes ses exigences.

Le planificateur de services envoie un *service-name* n'est pas en mesure de placer une tâche dans *Availability Zone* événement de service car aucune instance de conteneur ne répondait à toutes ses exigences. Pour résoudre ce problème, lancez des instances dans la zone de disponibilité.

service (*service-name*) ne parvient pas à placer une tâche dans *Zone de disponibilité*.

Le planificateur de services envoie un *service-name* n'est pas en mesure de placer une tâche dans *Availability Zone* événement de service lorsque vous utilisez le Fargate et qu'il n'y a pas de capacité disponible.

Vous pouvez ajouter des sous-réseaux supplémentaires dans la zone de disponibilité dans le message d'erreur, ou contacter Support pour obtenir une capacité supplémentaire.

service (*service-name*) n'a pas pu effectuer AZ Rebalance car *nom de l'ensemble de tâches* n'a pas pu évoluer en raison de *raison*.

Le planificateur de services envoie un *service-name* n'a pas pu effectuer AZ Rebalance car *task-set-name* n'a pas pu évoluer en raison de *reason* événement de service lorsque vous utilisez la protection évolutive des tâches.

Vous pouvez effectuer l'une des actions suivantes :

- Attendez que la protection des tâches en cours expire pour pouvoir y mettre fin.
- Déterminez les tâches qui peuvent être arrêtées et utilisez l'API `UpdateTaskProtection` avec l'option `protectionEnabled` définie sur `false` pour désactiver la protection de ces tâches.
- Augmentez le nombre de tâches souhaité pour le service à un nombre supérieur au nombre de tâches protégées.

service (*service-name*) a arrêté AZ Rebalancing.

Le planificateur de services envoie un *service-name* a arrêté l'événement du service AZ Rebalancing lorsque l'opération de rééquilibrage de la zone de disponibilité s'est arrêtée. Il s'agit d'un événement d'information. Amazon ECS envoie des événements supplémentaires qui fournissent davantage d'informations.

Résolution des problèmes liés aux équilibreurs de charge des services dans Amazon ECS

Les services Amazon ECS service peuvent enregistrer des tâches auprès d'un équilibreur de charge Elastic Load Balancing. Les erreurs de configuration d'équilibreur de charge sont des causes courantes de l'arrêt des tâches. Si vos tâches arrêtées ont été lancées par des services qui utilisent un équilibreur de charge, pensez aux causes possibles suivantes.

Le rôle lié à un service Amazon ECS n'existe pas

> Le rôle lié à un service Amazon ECS service permet aux services Amazon ECS service d'enregistrer des instances de conteneur avec des équilibreurs de charge Elastic Load Balancing. Le rôle lié à un service doit être créé dans votre compte. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.

Groupe de sécurité des instances de conteneur

Si votre conteneur est mappé au port 80 sur votre instance de conteneur, votre groupe de sécurité d'instance de conteneur doit autoriser le trafic entrant sur le port 80 pour que les surveillances de l'état de l'équilibreur de charge réussissent.

L'équilibreur de charge Elastic Load Balancing n'est pas configuré pour toutes les zones de disponibilité

Votre équilibreur de charge doit être configuré pour utiliser toutes les zones de disponibilité d'une région, ou au moins toutes les zones de disponibilité dans lesquelles se trouvent vos instances de conteneur. Si un service utilise un équilibreur de charge et lance une tâche sur une instance de conteneur qui réside dans une zone de disponibilité pour laquelle l'équilibreur de charge n'est pas configuré, la tâche en question ne réussira jamais la surveillance de l'état. La tâche est alors supprimée.

Le contrôle de santé de l'équilibreur de charge Elastic Load Balancing est mal configuré

Les paramètres de surveillance de l'état de l'équilibreur de charge peuvent être trop restrictifs ou pointer vers des ressources qui n'existent pas. Si une instance de conteneur n'est pas considérée comme saine, elle est supprimée de l'équilibreur de charge. Veillez à vérifier si les paramètres suivants sont correctement configurés pour votre équilibreur de charge de service.

Ping Port

La valeur Ping Port pour une surveillance de l'état d'équilibreur de charge correspond au port des instances de conteneur vérifiées par l'équilibreur de charge pour en déterminer l'état. Si ce port n'est pas correctement configuré, l'équilibreur de charge est susceptible d'annuler l'enregistrement de votre instance de conteneur. Ce port doit être configuré de manière à utiliser la valeur `hostPort` correspondant au conteneur dans la définition de tâche de votre service que vous utilisez avec la surveillance de l'état.

Ping Path

Cela fait partie de la surveillance de l'état de l'équilibreur de charge. Il s'agit d'un point de terminaison de votre application qui peut renvoyer un code d'état valide (par exemple, 200) lorsque l'application est saine. Cette valeur est souvent définie sur `index.html`, mais si votre service ne répond pas à cette demande, la surveillance de l'état échoue. Si votre conteneur n'a pas de fichier `index.html`, vous pouvez définir la valeur `/` afin de cibler l'URL de base pour l'instance de conteneur.

Response Timeout

Cette valeur correspond au délai dont dispose votre conteneur pour renvoyer une réponse à la commande ping de surveillance de l'état. Si cette valeur est inférieure à la durée nécessaire pour une réponse, la surveillance de l'état échoue.

Health Check Interval

Il s'agit de la durée entre les pings de surveillance de l'état. Plus vos intervalles de surveillance de l'état sont courts, plus votre instance de conteneur peut atteindre rapidement le seuil de défectuosité.

Unhealthy Threshold

Il s'agit du nombre d'échecs possibles des surveillances de l'état avant que votre instance de conteneur ne soit considérée comme étant défectueuse. Si vous avez un seuil de défectuosité de 2 et un intervalle de surveillance de l'état de 30 secondes, votre tâche a 60 secondes pour répondre au ping de surveillance de l'état avant d'être considérée comme défectueuse. Vous pouvez augmenter le seuil de défectuosité ou l'intervalle de surveillance de l'état afin de donner à vos tâches plus de temps pour répondre.

Impossible de mettre à jour le service *`servicename`*: le nom ou le port du conteneur de l'équilibreur de charge a été modifié dans la définition de la tâche

Si votre service utilise un équilibreur de charge, vous pouvez utiliser AWS CLI ou SDK pour modifier la configuration de l'équilibreur de charge. Pour plus d'informations sur la façon de modifier la configuration, voir UpdateService dans le Référence de l'API Amazon Elastic Container Service. Si vous mettez à jour la définition de tâche pour le service, le nom et le port de conteneur spécifiés dans la configuration de l'équilibreur de charge doivent rester dans la définition de tâche.

Vous avez atteint la limite du nombre de tâches que vous pouvez exécuter simultanément.

Pour un nouveau compte, vos quotas peuvent être inférieurs aux Service Quotas. Le quota de service de votre compte peut être affiché dans la console Service Quotas. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

Résolution des problèmes d’autoscaling du service dans Amazon ECS

L’autoscaling de l’application désactive les processus de réduction horizontale pendant que les déploiements Amazon ECS sont en cours. Ces processus reprennent une fois le déploiement terminé. Toutefois, pendant les déploiements, les processus de montée en puissance se poursuivent, sauf s'ils sont suspendus. Pour de plus amples informations, veuillez consulter Suspension et reprise de la mise à l'échelle pour Application Auto Scaling.

Capture d'événements Amazon ECS dans la console

La console Amazon ECS fournit une fonctionnalité de capture d'événements qui stocke Amazon ECS-generated événements, tels que les actions de service et les changements d'état des tâches, destinés à Amazon CloudWatch Se connecte EventBridge. Cette fonctionnalité inclut une interface d’interrogation dotée de filtres pour la surveillance et la résolution des problèmes.

Les événements fournissent des informations détaillées sur le fonctionnement de vos déploiements de service, services, tâches et instances. Vous pouvez utiliser ces informations pour résoudre les problèmes de déploiement de vos tâches et services.

Lorsque vous activez la capture d'événements, vous avez accès à tous les événements générés par Amazon ECS pendant la période de conservation de votre choix, allant au-delà des limites natives des 100 derniers événements non filtrés ou des tâches interrompues visibles pendant une heure seulement.

Comment ça marche

Utilisations de la capture d'événements EventBridge pour stocker des événements dans un Amazon prédéfini CloudWatch Groupe de journaux. La console Amazon ECS fournit des requêtes prédéfinies et des options de filtrage, et met en corrélation les événements afin de présenter les cycles de vie des tâches dans un format intuitif.

Vous pouvez interroger et récupérer les types d’événements suivants :

- Événements liés aux actions de service : permettent d’identifier les problèmes de provisionnement ou d’allocation des ressources
- Événements liés au cycle de vie des tâches : permettent d’identifier les raisons pour lesquelles les tâches ou les conteneurs ne parviennent pas à démarrer ou cessent de fonctionner.

La console Amazon ECS vous permet de configurer la capture d'événements en un clic et fournit les requêtes et le filtrage les plus courants sans que vous ayez à apprendre les langages des requêtes ou à naviguer entre plusieurs consoles.

Types d’événements

La capture des événements stocke tous les événements générés par Amazon ECS dans les catégories suivantes :

Événements de modification de l'état de la tâche

Les arrêts de conteneurs et autres événements de terminaison, que vous pouvez utiliser pour résoudre des problèmes ou pour surveiller le calendrier du cycle de vie des tâches.

Actions de service

Atteinte d’un état stable, échec d’un placement de tâche, contraintes en matière de ressources, etc.

Changement d’état du déploiement du service

Des événements tels que des déploiements en cours, terminés ou ayant échoué, déclenchés par des paramètres de disjoncteur et de restauration, pour surveiller l'état du déploiement d'un service.

Changement d’état des instances de conteneur

Pour les charges de travail sur les instances gérées EC2 et Amazon ECS, les événements indiquent l'état de connexion et de déconnexion.

Configuration du groupe de journaux

Lorsque vous activez la capture d'événements, Amazon ECS crée automatiquement les ressources suivantes :

- Une Amazon CloudWatch Groupe de journaux nommé `/aws/events/ecs/containerinsights/${clusterName}/performance`
- Un EventBridge règle qui ingère tous les événements provenant du `aws.ecs` source et les transmet au groupe de journaux

Vous pouvez spécifier une période de conservation pour le groupe de journaux allant de 1 jour à 10 ans. La période de conservation par défaut est de 7 jours.

Considérations

Lorsque vous utilisez la capture d'événements, tenez compte des points suivants :

- La capture d'événements enregistre tous les événements pour plus de simplicité. Vous ne pouvez pas configurer de règles dans la console Amazon ECS pour capturer uniquement des événements spécifiques.
- La console Amazon ECS fournit des critères de requête prédéfinis. Pour les requêtes avancées, utilisez Amazon CloudWatch Logs Logs Insights pour interroger directement le groupe de journaux.
- La fonctionnalité Live Tail n'est pas disponible dans la console Amazon ECS. Utilisez Amazon CloudWatch Se connecte directement à Live Tail.
- Lorsque vous désactivez la capture d'événements, EventBridge la règle est supprimée.
- La capture d'événements entraîne des coûts supplémentaires pour EventBridge ingestion de données, Amazon CloudWatch Stockage des journaux et exécution des requêtes.

  Pour plus d'informations sur EventBridge tarification, voir EventBridge tarification.

  Pour plus d'informations sur CloudWatch tarification, voir CloudWatch tarification.

Event-based dépannage

Utilisez les événements générés par Amazon ECS pour répondre aux questions de dépannage courantes.

Analyse des échecs de tâches

Vous pouvez revoir `STOPPED` événements de modification de l'état des tâches, codes d'arrêt et codes de sortie du conteneur pour déterminer pourquoi une tâche n'a pas pu être lancée ou a échoué en cours d'exécution.

Vous pouvez consulter les événements d'action de service pour détecter les échecs de placement et les informations relatives aux contraintes de ressources afin de déterminer pourquoi une tâche n'a pas pu être exécutée en raison de contraintes de ressources.

Scénarios courants d'échec des tâches

Les échecs de tâches anormaux les plus courants sont liés aux problèmes suivants :

- CI/CD échecs de déploiement de services
- Pannes de dimensionnement automatique
- Echecs de rééquilibrage des tâches
- Des sorties de conteneurs anormales, telles que des erreurs de mémoire insuffisante (OOM)

Des échecs de tâches anormaux produisent `STOPPED` événements de changement d'état des tâches avec un `EssentialContainerExited` ou `TaskFailedToStart` code d'arrêt. Vous pouvez filtrer en fonction de ces codes d'arrêt pour examiner l'exécution des conteneurs et les comportements d'arrêt.

Activer la capture d'événements pour un cluster Amazon ECS existant

Vous pouvez activer la capture d'événements sur un cluster Amazon ECS existant pour stocker Amazon ECS-generated événements sur Amazon CloudWatch Se connecte EventBridge. Cette fonctionnalité vous permet de surveiller et de résoudre les échecs de tâches, les déploiements de services et les autres activités du cluster.

Après avoir activé la capture d'événements, Amazon ECS crée les ressources suivantes :

- Une Amazon CloudWatch Groupe de journaux nommé `/aws/events/ecs/containerinsights/${clusterName}/performance`
- Un EventBridge règle qui capture tous les événements du `aws.ecs` source

UNE L'histoire L'onglet s'affiche dans la vue du cluster, vous permettant d'interroger les événements du cycle de vie des tâches et les actions de service. La capture des événements commence immédiatement et stocke tous les produits Amazon ECS-generated événements en fonction de la période de conservation que vous avez spécifiée.

Conditions préalables

- Un cluster Amazon ECS existant
- Autorisations IAM appropriées pour modifier les paramètres du cluster et créer Amazon CloudWatch Ressources des journaux

Activer la capture d'événements à l'aide de la console

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Clusters.
3. Sélectionnez le cluster sur lequel vous souhaitez activer la capture d'événements.

   La page des détails du cluster s’ouvre.
4. Choisissez Configuration.
5. Dans le Événements ECS section, choisissez Activer la capture d'événements.

   Le Activer la capture d'événements une boîte de dialogue s'affiche.
6. Pour Événement d'expiration, choisissez la période de conservation pour Amazon CloudWatch Groupe de journaux. La valeur par défaut est de 7 jours.
7. Choisissez Activer.

Affichage des événements de changement d'état du service et des tâches Amazon ECS

La console Amazon ECS fournit une fonctionnalité de capture d'événements qui stocke Amazon ECS-generated événements, tels que les actions de service et les changements d'état des tâches, destinés à Amazon CloudWatch Se connecte EventBridge. Cette fonctionnalité inclut une interface de requête dotée de fonctionnalités de filtrage pour améliorer la surveillance et le dépannage.

Les événements fournissent des informations détaillées sur le fonctionnement de vos déploiements de service, services, tâches et instances. Vous pouvez utiliser ces informations pour résoudre les problèmes de déploiement de vos tâches et services.

Vous pouvez utiliser l'un des critères suivants pour filtrer les événements :

- ID de déploiement (disponible uniquement sur la page détaillée du service)
- L’heure de début
- L’heure de fin
- Nom du service (applicable uniquement sur la page de détail du cluster, sur la page de détail du service, il s'agira par défaut du service actuel)
- ID de tâche
- Dernier état de la tâche

- Famille de définitions de tâches
- Révision de la définition des tâches

Affichage des événements au niveau du cluster

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Choisissez Clusters.

   La page de liste des clusters s'affiche.
3. Choisissez le cluster .

   La page des détails du cluster s’ouvre.
4. En dessous L'histoire, déterminez les événements à afficher.

   a. Pour afficher les événements relatifs aux opérations de maintenance, choisissez Événements d'action de service.

   b. Pour afficher les événements de changement d'état des tâches, choisissez Événements de changement d'état des tâches.

   c. (Facultatif) Dans Critères de requête, entrez les filtres pour les événements que vous souhaitez consulter.
5. Choisissez Exécuter la requête.

   Les événements s'affichent dans une liste.
6. Pour consulter tous les détails de l'événement, choisissez l'événement.

Affichage au niveau du service

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster.
3. Sur la page des détails du cluster, dans la section Services, choisissez le service.

   La page des détails du service s’affiche.
4. En dessous L'histoire, déterminez les événements à afficher.

   a. Pour afficher les événements relatifs aux opérations de maintenance, choisissez Événements d'action de service.

b. Pour afficher les événements de changement d'état des tâches, choisissez Événements de changement d'état des tâches.

c. (Facultatif) Dans Critères de requête, entrez les filtres pour les événements que vous souhaitez consulter.

5. Choisissez Exécuter la requête.

   Les événements s'affichent dans une liste.

6. Pour consulter tous les détails de l'événement, choisissez l'événement.

Résolution des erreurs d'UC ou de mémoire non valides liées à la définition des tâches Amazon ECS

Lorsque vous enregistrez une définition de tâche à l'aide de l'API Amazon ECS ou AWS CLI, si vous indiquez une valeur non valide `cpu` ou `memory` valeur, l'erreur suivante est renvoyée.

```
An error occurred (ClientException) when calling the RegisterTaskDefinition operation:
 Invalid 'cpu' setting for task.
```

> **Note**
>
> Si vous utilisez Terraform, l'erreur suivante peut être renvoyée.
>
> ```
> Error: ClientException: No Fargate configuration exists for given values.
> ```

Pour résoudre ce problème, vous devez spécifier une valeur prise en charge pour l'UC et la mémoire de la tâche dans votre définition de tâche. La valeur `cpu` peut être exprimée en unités d'UC ou de vCPU dans une définition de tâche. Elle est convertie en un nombre entier indiquant le nombre d'unités d'UC lorsque la définition de la tâche est enregistrée. La valeur `memory` peut être exprimée en Mio ou en Go dans une définition de tâche. Elle est convertie en un entier indiquant le nombre de Mio lorsque la définition de la tâche est enregistrée.

Pour les définitions de tâches qui spécifient `FARGATE` pour le paramètre `requiresCompatibilities` (même si `EC2` est également spécifié), vous devez utiliser l'une des valeurs indiquées dans le tableau suivant. Ces valeurs déterminent votre plage de valeurs prises en charge pour le paramètre d'UC et de mémoire.

Pour les tâches hébergées sur Fargate, le tableau suivant indique les combinaisons de processeur et de mémoire valides. Les valeurs de mémoire du fichier JSON sont spécifiées en Mio. Vous pouvez convertir la valeur en Go en Mio en la multipliant par 1 024. Par exemple, 1 Go = 1 024 Mio.

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
256 (0,25 vCPU)	512 Mio, 1 Go, 2 Go	Linux
512 (0,5 vCPU)	1 Go, 2 Go, 3 Go, 4 Go	Linux
1 024 (1 vCPU)	2 Go, 3 Go, 4 Go, 5 Go, 6 Go, 7 Go, 8 Go	Linux, Windows
2 048 (2 vCPU)	Entre 4 Go et 16 Go par incréments de 1 Go	Linux, Windows
4 096 (4 vCPU)	Entre 8 Go et 30 Go par incréments de 1 Go	Linux, Windows
8192 (8 vCPU) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 16 Go et 60 Go par incréments de 4 Go	Linux
16384 (16 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	Entre 32 Go et 120 Go par incréments de 8 Go	Linux

Valeur d'UC	Valeur de mémoire	Systèmes d'exploitation pris en charge pour AWS Fargate
32768 (32 processeurs virtuels) **Note** Cette option nécessite la plateforme Linux `1.4.0` ou ultérieure	60 GO, 120 GO, 244 GO	Linux

Pour les tâches hébergées sur Amazon EC2, les valeurs d'UC de tâche prises en charge sont comprises entre 0,25 vCPU et 192 vCPU.

Le mécanisme de contrôle de l'UC diffère entre EC2 et Fargate :

- Pour les tâches hébergées sur Amazon EC2 : Amazon ECS utilise la période d'UC et le quota d'UC pour contrôler les limites strictes de taille des tâches en termes d'UC. Lorsque vous spécifiez le vCPU dans votre définition de tâche, Amazon ECS traduit la valeur en période d'UC et en paramètres de quota d'UC qui s'appliquent au `cgroup`.
- Pour les tâches hébergées sur Fargate : Amazon ECS utilise des partages d'UC pour contrôler l'allocation de l'UC. Les valeurs de quota et de période d'UC ne sont pas utilisées pour limiter l'UC dans les tâches Fargate.

Pour les tâches Amazon EC2, le quota d'UC contrôle le temps processeur accordé à un `cgroup` pendant une période d'UC donnée. Les deux paramètres sont exprimés en microsecondes. Lorsque le quota d'UC est égal à la période d'UC, cela signifie qu'un `cgroup` peut exécuter jusqu'à 100 % sur un vCPU (ou toute autre fraction totalisant 100 % pour plusieurs vCPU). Le quota d'UC est au maximum de 1 000 000 us et la période d'UC est d'au moins 1 ms. Vous pouvez utiliser ces valeurs pour définir les limites de votre nombre d'UC. Lorsque vous modifiez la période d'UC sans modifier le quota d'UC, vous disposez de limites effectives différentes de celles que vous avez spécifiées dans votre définition de tâche.

La période de 100 ms permet d'utiliser de 0,125 à 10 vCPU.

> **Note**
>
> Task-level Les paramètres du processeur et de la mémoire sont ignorés pour les conteneurs Windows.

Affichage du journal d'agent de conteneur Amazon ECS

Amazon ECS stocke les journaux dans le dossier `/var/log/ecs` de vos instances de conteneur. Des journaux sont disponibles auprès de l'agent de conteneur Amazon ECS et du `ecs-init` service qui contrôle l'état de l'agent (start/stop) sur l'instance du conteneur. Vous pouvez afficher ces fichiers journaux en vous connectant à une instance de conteneur à l'aide de SSH.

> **Note**
>
> Si vous n'êtes pas sûr de savoir comment collecter tous les journaux de vos instances de conteneur, vous pouvez utiliser le collecteur de journaux d'Amazon ECS. Pour plus d'informations, consultez Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS.

Système d'exploitation Linux

Le processus `ecs-init` stocke les journaux à l'emplacement `/var/log/ecs/ecs-init.log`.

Le fichier `ecs-init.log` contient des informations sur la gestion du cycle de vie, la configuration et l'amorçage de l'agent conteneur.

Vous pouvez utiliser la commande suivante pour visualiser les fichiers journaux.

```
cat /var/log/ecs/ecs-init.log
```

Sortie :

```
2018-02-16T18:13:54Z [INFO] pre-start
2018-02-16T18:13:56Z [INFO] start
2018-02-16T18:13:56Z [INFO] No existing agent container to remove.
2018-02-16T18:13:56Z [INFO] Starting Amazon Elastic Container Service Agent
```

Système d'exploitation Windows

Vous pouvez utiliser le collecteur de journaux Amazon ECS pour Windows. Pour plus d'informations, consultez la section Amazon ECS Logs Collector pour Windows sur Github.

1. Connectez-vous à votre instance.
2. Ouvrir PowerShell puis exécutez les commandes suivantes avec des privilèges administratifs. Les commandes téléchargent le script et collectent les journaux.

```
Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://
raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-
logs-collector.ps1
.\ecs-logs-collector.ps1
```

Vous pouvez activer la journalisation de débogage pour l'agent Amazon ECS et le démon Docker. Cette option permet au script de collecter les journaux avant d'activer le mode de débogage. Le script redémarre le démon Docker et l'agent Amazon ECS, puis met fin à tous les conteneurs exécutés sur l'instance. Avant d'exécuter la commande suivante, videz l'instance de conteneur et déplacez les tâches importantes vers d'autres instances de conteneur.

Exécutez la commande suivante pour activer la journalisation.

```
.\ecs-logs-collector.ps1 -RunMode debug
```

Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS

> **Note**
>
> Vous ne pouvez pas utiliser le collecteur de journaux Amazon ECS sur les instances gérées Amazon ECS.

Si vous n'êtes pas sûr de savoir comment de collecter les différents journaux de vos instances de conteneur, vous pouvez utiliser le collecteur de journaux d'Amazon ECS. Il est disponible sur GitHub pour les deux Linux et Fenêtres. Le script collecte les journaux généraux du système d'exploitation ainsi que les journaux des agents de conteneur Docker et Amazon ECS, ce qui peut être utile pour

le dépannage. AWS Support cas. Puis, il compresse et archive les informations collectées dans un seul fichier qui peut être facilement partagé à des fins de diagnostic. Il prend également en charge l'activation du mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS sur les variantes d'Amazon Linux, telles que Amazon ECS-optimized AMI.

> **Note**
>
> Sur Amazon Linux Amazon ECS-optimized AMI version 20250909 et versions ultérieures, le collecteur de journaux Amazon ECS est préinstallé sur `/opt/amazon/ecs/ecs-logs-collector.sh` et prêt à être utilisé sans téléchargement depuis GitHub. Pour plus d'informations, voir Collecteur de journaux ECS dans le ECS-optimized Documentation sur l'AMI.

Actuellement, le collecteur de journaux Amazon ECS prend en charge les systèmes d'exploitation suivants :

- Amazon Linux
- Utilisation de Red Hat Enterprise Linux
- Ubuntu
- Windows Server

Pour exécuter le collecteur de journaux Amazon ECS pour Linux (ECS-optimized AMI)

1. Connectez-vous à votre instance de conteneur.
2. Exécutez le script pour collecter les journaux et créer l'archive.

> **Note**
>
> Pour activer le mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS, ajoutez l'option `--mode=enable-debug` à la commande suivante. Cela pourrait redémarrer le démon Docker, ce qui entraînerait la suppression de tous les conteneurs en cours d'exécution sur l'instance. Pensez à drainer l'instance de conteneur et à transférer les tâches importantes vers d'autres instances de conteneur avant d'activer le mode de débogage. Pour plus d'informations, consultez Drainage des instances de conteneur Amazon ECS.

```
[ec2-user ~]$ sudo /opt/amazon/ecs/ecs-logs-collector.sh
```

Une fois que vous avez exécuté le script, vous pouvez examiner les journaux collectés dans le dossier `collect` créé par le script. Le `collect.tgz` le fichier est une archive compressée de tous les journaux, que vous pouvez partager avec AWS Support pour obtenir de l'aide au diagnostic.

Pour télécharger et exécuter le collecteur de journaux Amazon ECS pour Linux

1. Connectez-vous à votre instance de conteneur.
2. Téléchargez le script de collecteur de journaux Amazon ECS.

   ```
   curl -O https://raw.githubusercontent.com/awslabs/ecs-logs-collector/master/ecs-logs-collector.sh
   ```

3. Exécutez le script pour collecter les journaux et créer l'archive.

   ```
   $ sudo bash ./ecs-logs-collector.sh
   ```

Pour télécharger et exécuter le collecteur de journaux Amazon ECS pour Windows

1. Connectez-vous à votre instance de conteneur. Pour plus d'informations, consultez la section Connexion à votre instance Windows à l'aide de RDP dans le Guide de l'utilisateur Amazon EC2.
2. Téléchargez le script du collecteur de journaux Amazon ECS à l'aide de PowerShell.

   ```
   Invoke-WebRequest -OutFile ecs-logs-collector.ps1 https://raw.githubusercontent.com/awslabs/aws-ecs-logs-collector-for-windows/master/ecs-logs-collector.ps1
   ```

3. Exécutez le script pour collecter les journaux et créer l'archive.

   > **Note**
   >
   > Pour activer le mode de débogage pour le démon Docker et l'agent de conteneur Amazon ECS, ajoutez l'option `-RunMode debug` à la commande suivante. Cette action redémarre le démon Docker, ce qui supprime tous les conteneurs qui s'exécutent sur l'instance. Pensez à drainer l'instance de conteneur et à transférer les tâches

importantes vers d'autres instances de conteneur avant d'activer le mode de débogage. Pour plus d'informations, consultez Drainage des instances de conteneur Amazon ECS.

```
.\ecs-logs-collector.ps1
```

Une fois que vous avez exécuté le script, vous pouvez examiner les journaux collectés dans le dossier `collect` créé par le script. Le `collect.tgz` le fichier est une archive compressée de tous les journaux, que vous pouvez partager avec AWS Assistance pour l'aide au diagnostic.

Récupération des informations de diagnostic d'Amazon ECS avec l'introspection d'agent

L'API d'introspection d'agent Amazon ECS fournit des informations sur l'état général de l'agent Amazon ECS et des instances de conteneur.

Vous pouvez utiliser l'API d'introspection d'agent pour obtenir l'ID Docker d'un conteneur dans votre tâche. Vous pouvez aussi utiliser l'API d'introspection d'agent en vous connectant à une instance de conteneur à l'aide de SSH.

Important

Votre instance de conteneur doit avoir un rôle IAM qui autorise l'accès à Amazon ECS afin d'atteindre l'API d'introspection. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

L'exemple suivant illustre deux tâches : une en cours d'exécution et une qui a été arrêtée.

Note

La commande suivante est redirigée via python -mjson.tool pour une lecture plus facile.

```
curl http://localhost:51678/v1/tasks | python -mjson.tool
```

Sortie :

```
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  1095  100  1095    0     0   117k      0 --:--:-- --:--:-- --:--:--  133k
{
    "Tasks": [
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/090eff9b-1ce3-4db6-848a-
a8d14064fd24",
            "Containers": [
                {
                    "DockerId":
 "189a8ff4b5f04affe40e5160a5ffadca395136eb5faf4950c57963c06f82c76d",
                    "DockerName": "ecs-console-sample-app-static-6-simple-
app-86caf9bcabe3e9c61600",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "f7f1f8a7a245c5da83aa92729bd28c6bcb004d1f6a35409e4207e1d34030e966",
                    "DockerName": "ecs-console-sample-app-static-6-busybox-
ce83ce978a87a890ab01",
                    "Name": "busybox"
                }
            ],
            "Family": "console-sample-app-static",
            "KnownStatus": "STOPPED",
            "Version": "6"
        },
        {
            "Arn": "arn:aws:ecs:us-west-2:aws_account_id:task/1810e302-eaea-4da9-
a638-097bea534740",
            "Containers": [
                {
                    "DockerId":
 "dc7240fe892ab233dbbcee5044d95e1456c120dba9a6b56ec513da45c38e3aeb",
                    "DockerName": "ecs-console-sample-app-static-6-simple-app-
f0e5859699a7aecfb101",
                    "Name": "simple-app"
                },
                {
                    "DockerId":
 "096d685fb85a1ff3e021c8254672ab8497e3c13986b9cf005cbae9460b7b901e",
```

```
                    "DockerName": "ecs-console-sample-app-static-6-
busybox-92e4b8d0ecd0cce69a01",
                    "Name": "busybox"
                }
            ],
            "DesiredStatus": "RUNNING",
            "Family": "console-sample-app-static",
            "KnownStatus": "RUNNING",
            "Version": "6"
        }
    ]
}
```

Dans l'exemple précédent, la tâche arrêtée (*090eff9b-1ce3-4db6-848a-a8d14064fd24*) possède deux conteneurs. Vous pouvez utiliser docker inspect ***container-ID*** pour afficher des informations détaillées sur chaque conteneur. Pour plus d'informations, consultez Introspection de conteneur Amazon ECS.

Diagnostic Docker dans Amazon ECS

Docker fournit plusieurs outils de diagnostic qui peuvent vous aider à résoudre les problèmes que vous rencontrez avec vos conteneurs et vos tâches. Pour plus d'informations sur tous les utilitaires de ligne de commande Docker disponibles, consultez la Référence de la CLI Docker dans la documentation Docker. Vous pouvez accéder aux utilitaires de ligne de commande Docker en vous connectant à une instance de conteneur à l'aide de SSH.

Les codes de sortie fournis par les conteneurs Docker peuvent également offrir des informations de diagnostic (par exemple, le code de sortie 137 signifie que le conteneur a reçu un signal `SIGKILL`). Pour plus d'informations, consultez la rubrique Exit Status de la documentation Docker.

Recensement des conteneurs Docker dans Amazon ECS

Vous pouvez utiliser la commande docker ps de votre instance de conteneur pour dresser la liste des conteneurs en cours d'exécution. Dans l'exemple suivant, seul l'agent de conteneur Amazon ECS est en cours d'exécution. Pour plus d'informations, consultez la rubrique docker ps de la documentation Docker.

```
docker ps
```

Sortie :

```
CONTAINER ID        IMAGE                            COMMAND             CREATED
      STATUS              PORTS                          NAMES
cee0d6986de0        amazon/amazon-ecs-agent:latest   "/agent"            22 hours ago
      Up 22 hours         127.0.0.1:51678->51678/tcp   ecs-agent
```

Vous pouvez utiliser la commande docker ps -a pour afficher tous les conteneurs (même ceux qui ont été arrêtés ou supprimés). Cela peut s'avérer utile pour répertorier les conteneurs qui s'arrêtent inopinément. Dans l'exemple suivant, le conteneur `f7f1f8a7a245` a disparu il y a 9 secondes. Il n'est donc pas visible dans une sortie docker ps sans l'indicateur `-a`.

```
docker ps -a
```

Sortie :

```
CONTAINER ID        IMAGE                                              COMMAND
 CREATED             STATUS                          PORTS                          NAMES
db4d48e411b1        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  19 seconds ago                                                                     ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-c09288a6b0cba8a53700
f7f1f8a7a245        busybox:buildroot-2014.02                   "\"sh -c '/bin/sh -c
  22 hours ago        Exited (137) 9 seconds ago                                     ecs-
console-sample-app-static-6-busybox-ce83ce978a87a890ab01
189a8ff4b5f0        httpd:2                                     "httpd-foreground"
  22 hours ago        Exited (137) 40 seconds ago                                    ecs-
console-sample-app-static-6-simple-app-86caf9bcabe3e9c61600
0c7dca9321e3        amazon/ecs-emptyvolume-base:autogenerated   "not-applicable"
  22 hours ago                                                                       ecs-
console-sample-app-static-6-internalecs-emptyvolume-source-90fefaa68498a8a80700
cee0d6986de0        amazon/amazon-ecs-agent:latest              "/agent"
  22 hours ago        Up 22 hours                 127.0.0.1:51678->51678/tcp   ecs-
agent
```

Affichage des journaux Docker dans Amazon ECS

Vous pouvez consulter les flux `STDOUT` et `STDERR` d'un conteneur avec la commande docker logs. Dans cet exemple, les journaux sont affichés pour *`dc7240fe892a`* récipient et canalisé à travers le head commande pour des raisons de concision. Pour plus d'informations, consultez docker logs dans la documentation Docker.

> **Note**
>
> Les journaux Docker ne sont disponibles que sur l'instance de conteneur si vous utilisez le pilote de journal `json` par défaut. Si vous avez configuré vos tâches pour utiliser le `awslogs` log driver, alors les journaux de vos conteneurs sont disponibles dans CloudWatch Journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à CloudWatch.

```
docker logs dc7240fe892a | head
```

Sortie :

```
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
AH00558: httpd: Could not reliably determine the server's fully qualified domain name,
 using 172.17.0.11. Set the 'ServerName' directive globally to suppress this message
[Thu Apr 23 19:48:36.956682 2015] [mpm_event:notice] [pid 1:tid 140327115417472]
 AH00489: Apache/2.4.12 (Unix) configured -- resuming normal operations
[Thu Apr 23 19:48:36.956827 2015] [core:notice] [pid 1:tid 140327115417472] AH00094:
 Command line: 'httpd -D FOREGROUND'
10.0.1.86 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:48:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:29 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.0.154 - - [23/Apr/2015:19:49:50 +0000] "-" 408 -
10.0.1.86 - - [23/Apr/2015:19:49:58 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:49:59 +0000] "GET / HTTP/1.1" 200 348
10.0.1.86 - - [23/Apr/2015:19:50:28 +0000] "GET / HTTP/1.1" 200 348
10.0.0.154 - - [23/Apr/2015:19:50:29 +0000] "GET / HTTP/1.1" 200 348
time="2015-04-23T20:11:20Z" level="fatal" msg="write /dev/stdout: broken pipe"
```

Inspection des conteneurs Docker dans Amazon ECS

Si vous avez l'ID Docker d'un conteneur, vous pouvez l'examiner avec la commande docker inspect. L'inspection des conteneurs offre la vue la plus détaillée de l'environnement dans lequel un conteneur a été lancé. Pour plus d'informations, consultez la rubrique docker inspect de la documentation Docker.

```
docker inspect dc7240fe892a
```

Sortie :

```
[{
    "AppArmorProfile": "",
    "Args": [],
    "Config": {
        "AttachStderr": false,
        "AttachStdin": false,
        "AttachStdout": false,
        "Cmd": [
            "httpd-foreground"
        ],
        "CpuShares": 10,
        "Cpuset": "",
        "Domainname": "",
        "Entrypoint": null,
        "Env": [
            "PATH=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/
local/apache2/bin",
            "HTTPD_PREFIX=/usr/local/apache2",
            "HTTPD_VERSION=2.4.12",
            "HTTPD_BZ2_URL=https://www.apache.org/dist/httpd/httpd-2.4.12.tar.bz2"
        ],
        "ExposedPorts": {
            "80/tcp": {}
        },
        "Hostname": "dc7240fe892a",
...
```

Configuration de la sortie détaillée du démon Docker dans Amazon ECS

Si vous rencontrez des difficultés avec des conteneurs ou des images Docker, vous pouvez activer le mode de débogage sur votre démon Docker. L'utilisation du débogage fournit une sortie plus détaillée du démon. Vous pouvez l'utiliser pour récupérer les messages d'erreur envoyés depuis des registres de conteneurs, tels qu'Amazon ECR.

> **Important**
>
> Cette procédure est écrite pour Amazon ECS-optimized AMI Amazon Linux. Pour les autres systèmes d'exploitation, consultez les sections Activation du débogage et Contrôle et configuration de Docker avec systemd dans la documentation Docker.

Pour utiliser le mode de débogage du démon Docker sur Amazon ECS-optimized AMI Amazon Linux

1. Connectez-vous à votre instance de conteneur.
2. Ouvrez le fichier d'options Docker dans un éditeur de texte, tel que vi. Pour l'Amazon ECS-optimized Amazon Linux AMI, le fichier d'options Docker se trouve à `/etc/sysconfig/docker`.
3. Recherchez l'instruction d'options Docker, puis ajoutez l'option `-D` à la chaîne, à l'intérieur des guillemets.

   > **Note**
   >
   > Si l'instruction d'options Docker commence par le signe #, supprimez ce caractère pour supprimer la mise en commentaire de l'instruction et activer les options.

   Pour l'Amazon ECS-optimized Amazon Linux AMI, l'instruction d'options Docker s'appelle `OPTIONS`. Par exemple :

   ```
   # Additional startup options for the Docker daemon, for example:
   # OPTIONS="--ip-forward=true --iptables=true"
   # By default we limit the number of open files per container
   OPTIONS="-D --default-ulimit nofile=1024:4096"
   ```
4. Enregistrez le fichier et quittez votre éditeur de texte.
5. Redémarrez le démon Docker.

   ```
   sudo service docker restart
   ```

   La sortie est la suivante :

   ```
   Stopping docker:                                           [  OK  ]
   ```

```
Starting docker: .                                         [  OK  ]
```

6. Redémarrez l'agent Amazon ECS.

```
sudo service ecs restart
```

Vos journaux Docker doivent désormais présenter des informations plus détaillées.

```
time="2015-12-30T21:48:21.907640838Z" level=debug msg="Unexpected response from
 server: \"{\\\"errors\\\":[{\\\"code\\\":\\\"DENIED\\\",\\\"message\\\":\\\"User:
 arn:aws:sts::1111:assumed-role/ecrReadOnly/i-abcdefg is not authorized to perform:
 ecr:InitiateLayerUpload on resource: arn:aws:ecr:us-east-1:1111:repository/nginx_test
\\\"}]}\\n\" http.Header{\"Connection\":[]string{\"keep-alive\"}, \"Content-Type\":
[]string{\"application/json; charset=utf-8\"}, \"Date\":[]string{\"Wed, 30 Dec 2015
 21:48:21 GMT\"}, \"Docker-Distribution-Api-Version\":[]string{\"registry/2.0\"},
 \"Content-Length\":[]string{\"235\"}}"
```

Résoudre les problèmes liés au Docker **`Erreur d'API (500) : devmapper`** dans Amazon ECS

L'erreur Docker suivante indique que le stockage par groupe mince sur votre instance de conteneur est saturé et que le démon Docker ne peut pas créer de nouveaux conteneurs :

```
CannotCreateContainerError: API error (500): devmapper: Thin Pool has 4350 free data
 blocks which is less than minimum required 4454 free data blocks. Create more free
 space in thin pool or use dm.min_free_space option to change behavior
```

Par défaut, Amazon ECS-optimized AMI Amazon Linux à partir de la version `2015.09.d` et lancez-le ultérieurement avec un volume de 8 Go pour le système d'exploitation connecté à `/dev/xvda` et monté en tant que racine du système de fichiers. Un volume supplémentaire de 22 Gio est attaché à `/dev/xvdcz`, qui est utilisé par Docker pour le stockage des métadonnées et des images. Si cet espace de stockage est saturé, le démon Docker ne peut pas créer de nouveaux conteneurs.

La manière la plus simple d'ajouter du stockage à vos instances de conteneur est de résilier les instances existantes et d'en lancer de nouvelles avec de plus grands volumes de stockage de données. Toutefois, si vous ne pouvez pas procéder de cette façon, vous pouvez ajouter du stockage au groupe de volumes utilisé par Docker et étendre son volume logique en suivant les procédures indiquées dans Amazon ECS-optimized AMI Linux.

Si votre stockage d'instance de conteneur se remplit trop vite, vous pouvez limiter cela de la façon suivante :

- Pour afficher les informations par groupe mince, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker info
  ```

- (Agent de conteneur Amazon ECS 1.8.0 et version ultérieure) Vous pouvez réduire la durée pendant laquelle les conteneurs arrêtés ou fermés restent sur vos instances de conteneur. La variable de configuration de l'agent `ECS_ENGINE_TASK_CLEANUP_WAIT_DURATION` définit la durée d'attente entre l'arrêt d'une tâche et la suppression du conteneur Docker (par défaut, cette valeur est de 3 heures). Cette opération supprime les données du conteneur Docker. Si la valeur est trop basse, il est possible que vous ne puissiez pas examiner les conteneurs arrêtés ou afficher les journaux avant leur suppression. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.
- Vous pouvez supprimer les conteneurs qui ne sont pas en cours d'exécution et les images inutilisées de vos instances de conteneurs. Vous pouvez utiliser les exemples de commande suivants pour supprimer manuellement les conteneurs arrêtés et les images inutilisées. Les conteneurs supprimés ne peuvent pas être examinés par la suite et les images supprimées doivent être extraites à nouveau avant d'être utilisées pour lancer de nouveaux conteneurs.

  Pour supprimer les conteneurs qui ne sont pas en cours d'exécution, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker rm $(docker ps -aq)
  ```

  Pour supprimer les images non utilisées, exécutez la commande suivante sur votre instance de conteneur :

  ```
  docker rmi $(docker images -q)
  ```

- Vous pouvez supprimer les blocs de données inutilisés dans les conteneurs. Vous pouvez utiliser la commande suivante pour exécuter fstrim sur un conteneur en cours d'exécution et ignorer les blocs de données inutilisés par le système de fichiers du conteneur.

  ```
  sudo sh -c "docker ps -q | xargs docker inspect --format='{{ .State.Pid }}' | xargs -IZ fstrim /proc/Z/root/"
  ```

Résolution des problèmes liés à Amazon ECS Exec

Voici des notes de résolution pour vous aider à diagnostiquer d'où viennent les erreurs lorsque vous utilisez ECS Exec.

Vérification à l’aide d’Exec Checker

Le script ECS Exec Checker permet de vérifier que votre cluster et votre tâche Amazon ECS satisfont aux conditions préalables à l’utilisation de la fonctionnalité ECS Exec. Le script ECS Exec Checker vérifie à la fois votre AWS CLI l'environnement, le cluster et les tâches sont prêts pour ECS Exec, en appelant différentes API en votre nom. L'outil nécessite la dernière version du AWS CLI et que le `jq` est disponible. Pour plus d'informations, voir Vérificateur ECS Exec sur GitHub.

Erreur lors de l'appel **`Exécuter la commande`**

Si une erreur `The execute command failed` se produit, les éléments suivants peuvent en être à l'origine.

- La tâche ne dispose pas des autorisations requises. Vérifiez que la définition de tâche utilisée pour lancer votre tâche a un rôle IAM de tâche défini et que le rôle dispose des autorisations requises. Pour plus d’informations, consultez Autorisations ECS Exec.
- L’agent SSM n’est ni installé ni en cours d’exécution.
- Il existe un point de terminaison Amazon VPC d’interface pour Amazon ECS, mais il n’y en a pas pour le gestionnaire de session Systems Manager.

Résolution des problèmes liés à Amazon ECS Anywhere

Amazon ECS Anywhere fournit une assistance pour l’enregistrement d’une instance externe, telle qu’un serveur sur site ou une machine virtuelle (VM), sur votre cluster Amazon ECS. Voici les problèmes courants que vous pouvez rencontrer et les recommandations générales pour leur résolution.

Rubriques

- Problèmes d'enregistrement d'instance externe
- Problèmes de réseau d'instance externe
- Problèmes d'exécution de tâches sur votre instance externe

Problèmes d'enregistrement d'instance externe

Lorsque vous enregistrez une instance externe auprès de votre cluster Amazon ECS, les conditions suivantes doivent être remplies :

- Un AWS Systems Manager activation, qui consiste en un ID d'activation et code d'activation, doit être récupéré. Vous l'utilisez pour enregistrer l'instance externe en tant qu'instance gérée par Systems Manager. Lorsqu'une activation de Systems Manager est demandée, spécifiez une limite d'enregistrement et une date d'expiration. La limite d'enregistrement spécifie le nombre maximal d'instances qui peuvent être enregistrées à l'aide de l'activation. La valeur par défaut pour la limite d'enregistrement est de 1 instance. La date d'expiration spécifie la date à laquelle l'activation expire. La valeur par défaut est 24 heures. Si l'activation de Systems Manager que vous utilisez pour enregistrer votre instance externe n'est pas valide, demandez-en une nouvelle. Pour plus d'informations, consultez Enregistrement d'une instance externe dans un cluster Amazon ECS.
- Une politique IAM est utilisée pour fournir à votre instance externe les autorisations dont elle a besoin pour communiquer avec AWS Opérations d'API. Si cette stratégie gérée n'est pas créée correctement et ne contient pas les autorisations requises, l'enregistrement d'instance externe échoue. Pour plus d'informations, consultez Rôle IAM Amazon ECS Anywhere.
- Amazon ECS fournit un script d'installation qui installe Docker, l'agent de conteneur Amazon ECS et le Systems Manager Agent sur votre instance externe. Si le script d'installation échoue, il est probable qu'il ne puisse plus être exécuté sur la même instance sans qu'une erreur ne se produise. Dans ce cas, suivez le processus de nettoyage pour effacer AWS ressources de l'instance afin que vous puissiez exécuter à nouveau le script d'installation. Pour plus d'informations, consultez Annulation de l'enregistrement d'une instance externe Amazon ECS.

  > **Note**
  >
  > Sachez que si le script d'installation a demandé et utilisé avec succès l'activation de Systems Manager, toute autre exécution du script d'installation utilise à nouveau l'activation de Systems Manager. Cela peut à son tour vous amener à atteindre la limite d'enregistrement pour cette activation. Si cette limite est atteinte, vous devez recréer une activation.

- Lorsque vous exécutez le script d'installation sur une instance externe pour des charges de travail GPU, si le pilote NVIDIA n'est pas détecté ou configuré correctement, une erreur se produit. Le script d'installation utilise la commande `nvidia-smi` pour confirmer l'existence du pilote NVIDIA.

Problèmes de réseau d'instance externe

Pour communiquer toute modification, votre instance externe nécessite une connexion réseau à AWS. Si votre instance externe perd sa connexion réseau à AWS, les tâches qui s'exécutent sur vos instances continuent de s'exécuter malgré tout, sauf si elles sont arrêtées manuellement. Après la connexion à AWS est restauré, le AWS les informations d'identification utilisées par l'agent de conteneur Amazon ECS et l'agent Systems Manager sur l'instance externe sont renouvelées automatiquement. Pour plus d'informations sur le AWS domaines utilisés pour la communication entre votre instance externe et AWS, voir Mise en réseau.

Problèmes d'exécution de tâches sur votre instance externe

Si vos tâches ou conteneurs ne parviennent pas à s'exécuter sur votre instance externe, cela est généralement dû au réseau ou aux autorisations. Si vos conteneurs extraient leurs images d'Amazon ECR ou sont configurés pour envoyer des journaux de conteneurs à CloudWatch Journaux, votre définition de tâche doit spécifier un rôle IAM d'exécution de tâches valide. Sans un rôle IAM d'exécution de tâche valide, vos conteneurs ne démarrent pas. Pour en savoir plus sur les problèmes liés au réseau, consultez Problèmes de réseau d'instance externe.

> **Important**
>
> Amazon ECS fournit l'outil de collecte des journaux Amazon ECS. Vous pouvez l'utiliser pour collecter des journaux de vos instances externes à des fins de dépannage. Pour plus d'informations, consultez Collecte des journaux de conteneur avec collecteur de journaux Amazon ECS.

Résolution des problèmes de chargement des classes Java sur Fargate

Les applications Java exécutées sur Fargate peuvent rencontrer des problèmes de chargement de classes après les mises à jour de la plateforme, en particulier lorsque l'application utilise un comportement de chargement de classe non déterministe. Cela peut se manifester par des erreurs d'injection de dépendances, des échecs de Spring Boot ou d'autres exceptions d'exécution qui n'étaient pas présentes dans les déploiements précédents.

Symptômes

Vous pourriez observer les symptômes suivants :

- Erreurs d’injection de dépendance avec Spring Boot
- ClassNotFoundException ou NoClassDefFoundError exceptions
- Les applications qui fonctionnaient auparavant sur Fargate échouent désormais par intermittence
- La même image de conteneur fonctionne sur Amazon EC2, mais échoue sur Fargate
- Comportement incohérent entre les déploiements avec des images de conteneur identiques

Causes

Ces problèmes se produisent généralement pour les raisons suivantes :

- Non-deterministic chargement de classe : Les applications Java qui dépendent de l'ordre dans lequel les classes sont chargées à partir des fichiers JAR peuvent échouer lorsque la plate-forme sous-jacente modifie la façon dont les fichiers sont accessibles ou mis en cache.
- Mises à jour de la plateforme : les mises à jour de la version de plateforme Fargate peuvent modifier le comportement du système de fichiers sous-jacent, affectant ainsi l’ordre dans lequel les classes sont découvertes et chargées.
- Dépendances relatives à l’ordre des fichiers JAR : applications qui s’appuient implicitement sur des séquences de chargement JAR spécifiques sans gestion explicite des dépendances.

Résolution

Pour résoudre les problèmes de chargement de classes Java sur Fargate, implémentez des pratiques de chargement de classes déterministes :

Correction immédiate

Si vous avez besoin d’une solution immédiate :

1. Imposition de l’ordre de chargement des fichiers JAR : spécifiez explicitement l’ordre dans lequel les fichiers JAR doivent être chargés dans la configuration du chemin de classe de votre application.

2. Utilisation d'une gestion explicite des dépendances : assurez-vous que toutes les dépendances sont explicitement déclarées dans votre configuration de build (Maven, Gradle, etc.) plutôt que de vous fier à des dépendances transitives.

Long-term meilleures pratiques

Mettez en œuvre les pratiques suivantes pour éviter les futurs problèmes de chargement des classes :

1. Application d'un chargement des classes déterministe :
   - Utilisez des déclarations de dépendance explicites dans vos fichiers de construction
   - Évitez de vous fier à l'ordre d'analyse des chemins de classe
   - Utiliser des outils de gestion des dépendances pour résoudre les conflits de versions
   - Utilisez les options de la machine virtuelle Java (JVM), par exemple `-verbose:class` pour obtenir des informations sur les classes chargées par la machine virtuelle Java.
2. Applications Spring Boot :
   - Utilisation `@ComponentScan` avec des packages de base explicites
   - Évitez les conflits de configuration automatique en configurant explicitement les beans
   - Utiliser des annotations `@DependsOn` pour contrôler l'ordre d'initialisation des beans
3. Configuration de compilation :
   - Utiliser les sections de gestion des dépendances dans Maven ou Gradle
   - Exclure les dépendances transitives qui provoquent des conflits
   - Utilisez des outils tels que le plug-in Maven Enforcer pour détecter les problèmes de dépendance
4. Test :
   - Testez votre application avec différentes implémentations de JVM
   - Exécutez des tests d'intégration qui simulent différents environnements de déploiement
   - Utiliser des outils pour analyser les conflits de chemins de classe pendant le développement

Prévention

Pour éviter les problèmes de chargement des classes Java lors de futurs déploiements, procédez comme suit :

- Suivez les pratiques déterministes de chargement des classes : concevez votre application de manière à ce qu'elle ne dépende pas de l'ordre dans lequel les classes sont chargées à partir du chemin de classe.
- Utilisez une gestion explicite des dépendances : déclarez toujours explicitement toutes les dépendances requises et leurs versions dans votre configuration de compilation.
- Testez plusieurs environnements : testez régulièrement vos applications dans différents environnements et versions de plateforme afin d'identifier rapidement les problèmes potentiels.
- Surveillez les mises à jour de la plateforme : restez informé des mises à jour de la plateforme Fargate et testez vos applications avec les nouvelles versions de la plateforme avant qu'elles n'affectent les charges de travail de production.

AWS Fargate limitation des quotas

AWS Fargate limite les taux de lancement des tâches Amazon ECS et des pods Amazon EKS à des quotas (anciennement appelés limites) à l'aide d'un algorithme de bucket de jetons pour chaque AWS compte par région. Avec cet algorithme, votre compte dispose d'un compartiment contenant un nombre spécifique de jetons. Le nombre de jetons dans le compartiment représente votre quota de taux à une seconde donnée. Chaque compte client dispose d'un compartiment de jetons de tâches et de pods qui s'épuise en fonction du nombre de tâches et de pods lancés par le compte client. Ce compartiment de jetons affiche un maximum de compartiment qui vous permet d'effectuer périodiquement un nombre plus élevé de demandes, et un taux de recharge qui vous permet de maintenir un taux de demande stable aussi longtemps que nécessaire.

Par exemple, la taille du compartiment de jetons de tâches et de pods pour un compte client Fargate est de 100 jetons et le taux de recharge est de 20 jetons par seconde. Par conséquent, vous pouvez lancer immédiatement jusqu'à 100 tâches Amazon ECS et pods Amazon EKS par compte client, avec un taux de lancement soutenu de 20 tâches Amazon ECS et pods Amazon EKS par seconde.

Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Quota de taux de ressources Fargate pour les tâches Amazon ECS à la demande et les pods Amazon EKS[1]	100	20

Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Quota de taux de ressources Fargate pour les tâches Spot Amazon ECS	100	20

[1]Les comptes qui lancent uniquement des pods Amazon EKS ont un taux en rafale de 20 avec un taux de lancement soutenu de 20 pods par seconde lors de l'utilisation des versions de plateforme décrites dans les Versions de la plateforme Amazon EKS.

Limitation de l'API **RunTask** dans Fargate

En outre, Fargate limite le taux de requêtes lors du lancement de tâches à l'aide de l'API `RunTask` d'Amazon ECS à l'aide d'un quota distinct. Fargate limite Amazon ECS `RunTask` Demandes d'API pour chacun AWS compte par région. Chaque requête que vous effectuez supprime un jeton du compartiment. Le but est de favoriser la performance du service et d'assurer une utilisation équitable pour tous les clients Fargate. Les appels d'API sont soumis aux quotas de requêtes, qu'ils proviennent de la console Amazon Elastic Container Service, d'un outil de ligne de commande ou d'une application tierce. Le quota de taux pour les appels à l'API `RunTask` d'Amazon ECS est de 20 appels par seconde (rafale et soutenu). Chaque appel à cette API peut cependant lancer jusqu'à 10 tâches. Cela signifie que vous pouvez lancer 100 tâches en une seconde en effectuant 10 appels à cette API, demandant que 10 tâches soient lancées dans chaque appel. De même, vous pouvez également effectuer 20 appels à cette API, demandant que 5 tâches soient lancées dans chaque appel. Pour plus d'informations sur la limitation d'API pour l'API `RunTask` d'Amazon ECS, consultez la section Limitation des requêtes d'API dans la Référence de l'API Amazon ECS.

Dans la pratique, les taux de lancement des tâches et des pods dépendent également d'autres considérations telles que les images de conteneur à télécharger et à décompresser, les surveillances de l'état et d'autres intégrations activées, telles que l'enregistrement des tâches ou des pods auprès d'un équilibreur de charge. Les clients constatent des variations dans les taux de lancement des tâches et des pods par rapport aux quotas indiqués précédemment, en fonction des fonctionnalités qu'ils activent.

Ajustement des quotas de taux dans Fargate

Vous pouvez demander une augmentation des quotas de limitation de taux pour votre compte AWS . Pour plus d'informations, consultez Demande d'augmentation de quotas dans le Guide de l'utilisateur Service Quotas .

Résolution des problèmes liés aux instances gérées Amazon ECS

Utilisez les procédures suivantes pour dépanner les instances gérées Amazon ECS, notamment les problèmes courants, les techniques de diagnostic et les étapes de résolution.

Conditions préalables

Avant de résoudre les problèmes liés aux instances gérées Amazon ECS, assurez-vous de respecter les exigences suivantes.

- Le AWS CLI est installé et configuré avec les autorisations appropriées

  Pour plus d'informations, consultez la section Installation ou mise à jour vers la version la plus récente de l' AWS Command Line Interface dans le Guide de l'utilisateur AWS Command Line Interface .
- Accès à un cluster avec le fournisseur de capacité Amazon ECS Managed Instances. Pour plus d'informations, consultez the section called “Création d'un cluster pour les instances gérées Amazon ECS”.

Scénarios courants de résolution des problèmes

Affichage des journaux des agents de conteneurs Amazon ECS Managed Instances

Vous pouvez consulter ces fichiers journaux Amazon ECS dans les instances gérées Amazon ECS en vous connectant à un conteneur privilégié qui s'exécute dans l'instance.

Étapes diagnostiques

Déployez un conteneur de débogage doté de privilèges et de fonctionnalités Linux en tant que tâche Amazon ECS :

Définissez les variables d'environnement suivantes.

Remplacez le *user-input* avec tes valeurs.

```
export ECS_CLUSTER_NAME="your-cluster-name"
export AWS_REGION="your-region"
export ACCOUNT_ID="your-account-id"
```

Créez une définition de tâche à l'aide d'un fichier JSON de la CLI appelé `node-debugger.json`.

```
cat << EOF > node-debugger.json
{
  "family": "node-debugger",
  "taskRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
  "executionRoleArn": "arn:aws:iam::${ACCOUNT_ID}:role/ecsTaskExecutionRole",
  "cpu": "256",
  "memory": "1024",
  "networkMode": "host",
  "pidMode": "host",
  "requiresCompatibilities": ["MANAGED_INSTANCES", "EC2"],
  "containerDefinitions": [
    {
      "name": "node-debugger",
      "image": "public.ecr.aws/amazonlinux/amazonlinux:2023",
      "essential": true,
      "privileged": true,
      "command": ["sleep", "infinity"],
      "healthCheck": {
          "command": ["CMD-SHELL", "echo debugger || exit 1"],
          "interval": 30,
          "retries": 3,
          "timeout": 5
      },
      "linuxParameters": {
        "initProcessEnabled": true
      },
      "mountPoints": [
        {
          "sourceVolume": "host-root",
          "containerPath": "/host",
          "readOnly": false
        }
      ],
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/aws/ecs/node-debugger",
```

```
          "awslogs-create-group": "true",
          "awslogs-region": "${AWS_REGION}",
          "awslogs-stream-prefix": "ecs"
        }
      }
    }
  ],
  "volumes": [
    {
      "name": "host-root",
      "host": {
        "sourcePath": "/"
      }
    }
  ]
}
EOF
```

Inscrivez-vous, puis exécutez la tâche. Exécutez les commandes suivantes.

```
aws ecs register-task-definition --cli-input-json file://node-debugger.json

TASK_ARN=$(aws ecs run-task \
  --cluster $ECS_CLUSTER_NAME \
  --task-definition node-debugger \
  --enable-execute-command \
  --capacity-provider-strategy capacityProvider=managed-instances-default,weight=1 \
  --query 'tasks[0].taskArn' --output text)

# Wait for task to be running
aws ecs wait tasks-running --cluster $ECS_CLUSTER_NAME --tasks $TASK_ARN
```

Connectez-vous au conteneur. Exécutez la commande suivante.

```
aws ecs execute-command \
  --cluster $ECS_CLUSTER_NAME \
  --task $TASK_ARN \
  --container node-debugger \
  --interactive \
  --command "/bin/sh"
```

Consultez les journaux des agents Amazon ECS :

Dans la session interactive du conteneur, exécutez les commandes suivantes :

```
# Install required tools
yum install -y util-linux-core

# View ECS agent logs
nsenter -t 1 -m -p cat /var/log/ecs/ecs-agent.log | tail -50

# Check agent registration
nsenter -t 1 -m -p grep "Registered container instance" /var/log/ecs/ecs-agent.log

Example Output:

{"level":"info","time":"2025-10-16T12:39:37.665","msg":"Registered container instance
 with cluster!"}

# Verify capabilities
nsenter -t 1 -m -p grep "Response contained expected value for attribute" /var/log/ecs/
ecs-agent.log
```

Vérifiez les statistiques des agents :

Exécutez la commande suivante pour afficher les journaux.

```
# View metrics logs
nsenter -t 1 -m -p cat /var/log/ecs/metrics.log | tail -20
```

Problèmes de placement des tâches

Les symptômes des problèmes de placement des tâches sont les suivants :

- Tâches bloquées dans l'état EN ATTENTE
- Les tâches ne démarrent pas sur les instances gérées Amazon ECS
- Erreurs liées au manque de ressources

Étapes diagnostiques

Exécutez les commandes suivantes pour diagnostiquer les problèmes de placement des tâches et recueillir des informations sur la capacité du cluster, les instances de conteneurs et les services système :

```
# Check cluster capacity
aws ecs describe-clusters --clusters cluster-name --include STATISTICS

# Check cluster capacity providers
aws ecs describe-clusters --clusters cluster-name --include STATISTICS --query
 'clusters[].capacityProviders'

# List container instances
aws ecs list-container-instances --cluster cluster-name

# Check container instance details
aws ecs describe-container-instances --cluster cluster-name --container-
instances container-instance-arn

# Check container instance remaining resources CPU/Mem
aws ecs describe-container-instances --cluster $ECS_CLUSTER_NAME --container-
instances container-instance-arn --query 'containerInstances[].remainingResources'

# Check container instance Security Group
aws ecs describe-container-instances --cluster $ECS_CLUSTER_NAME --container-
instances container-instance-arn --query 'containerInstances[].ec2InstanceId' --output
 text
aws ec2 describe-instances --instance-ids instance-id --query
 'Reservations[0].Instances[0].SecurityGroups'
aws ec2 describe-security-groups --group-ids security-group-id
```

Surveillance des services du système :

```
# Check Containerd status
nsenter -t 1 -m -p systemctl status containerd.service

# Check Amazon ECS container agent status
nsenter -t 1 -m -p systemctl status ecs
```

Résolution

Pour résoudre les problèmes de placement des tâches, procédez comme suit pour garantir une configuration et une capacité appropriées :

- Vérifiez les besoins en ressources des tâches par rapport à la capacité disponible
- Vérifiez les contraintes et les stratégies de placement
- Assurez-vous que le fournisseur de capacité Amazon ECS Managed Instances est configuré

- Assurez-vous que le groupe de sécurité des tâches et des instances de conteneur dispose d'une règle sortante qui autorise le trafic vers les points de terminaison de gestion des agents Amazon ECS

Problèmes de mise en réseau

Les symptômes des problèmes de mise en réseau sont les suivants :

- Tâches impossibles à atteindre des services externes
- Problèmes de résolution DNS

Étapes diagnostiques

Tests de connectivité réseau :

À partir du conteneur de débogage, exécutez les commandes suivantes :

> **Note**
>
> Vérifiez que le groupe de sécurité associé à votre fournisseur de capacité ou à la tâche Amazon ECS autorise le trafic.

```
# Install DNS Utility
yum install bind-utils -y

# Test DNS resolution
nslookup amazon.com

# Test external connectivity
curl -I https://amazon.com
```

Contraintes liées aux ressources

Les symptômes des problèmes de mise en réseau sont les suivants :

- Tâches annulées en raison de limites de mémoire
- Limitation du processeur
- Problèmes d'espace disque

Étapes diagnostiques

Exécutez des commandes pour surveiller les limites de ressources et de conteneurs.

Surveillance des ressources :

```
# Check memory usage
nsenter -t 1 -m -p free -h

# Check disk usage
nsenter -t 1 -m -p lsblk

# Check disk usage
nsenter -t 1 -m -p df -h
```

Limites de conteneurs :

```
# Check OOM kills
nsenter -t 1 -m -p dmesg | grep -i "killed process"
```

Problème de déconnexion de l'agent d'instance de conteneur

Les symptômes des problèmes de déconnexion de l'agent d'instance de conteneur sont les suivants :

- Instances de conteneur affichées comme étant déconnectées dans la console Amazon ECS
- Les tâches ne peuvent pas être placées sur des instances spécifiques
- Échec de l'enregistrement des agents dans les journaux

Étapes diagnostiques

S'il existe une tâche de privilèges en cours d'exécution sur l'hôte à laquelle ECS Exec peut accéder, exécutez les commandes suivantes pour diagnostiquer les problèmes de connectivité des agents :

```
# check service status
nsenter -t 1 -m -p systemctl status ecs
nsenter -t 1 -m -p systemctl status containerd

# restart stopped services
nsenter -t 1 -m -p systemctl restart ecs
nsenter -t 1 -m -p systemctl restart containerd
```

Sinon, forcez le désenregistrement des instances gérées Amazon ECS. Exécutez la commande suivante :

```
# list ECS Managed Instance container
aws ecs list-container-instances --cluster managed-instances-cluster --query
 'containerInstanceArns' --output text

# deregister the specific container instance
aws ecs deregister-container-instance \
    --cluster $ECS_CLUSTER_NAME \
    --container-instance container-instance-arn \
    --force
```

Résolution

Pour résoudre les problèmes de déconnexion des agents, procédez comme suit :

- Vérifiez les autorisations de rôle IAM pour l'instance de conteneur
- Vérifiez que les règles des groupes de sécurité autorisent le trafic HTTPS sortant vers les terminaux ECS
- Garantissez la connectivité réseau à AWS services
- Redémarrez le service d'agent ECS si nécessaire : `nsenter -t 1 -m -p systemctl restart ecs`
- Vérifiez la configuration ECS_CLUSTER dans/etc/ecs/ecs.config correspond au nom de votre cluster

Analyse des journaux dans les instances gérées Amazon ECS

Journaux du système

Utilisez les commandes suivantes pour examiner les journaux système et identifier les problèmes potentiels liés à l'instance gérée :

```
# Check system messages
nsenter -t 1 -m -p journalctl --no-pager -n 50

# Check kernel logs
nsenter -t 1 -m -p dmesg | tail -20
```

```
# Check for disk space errors
nsenter -t 1 -m -p journalctl --no-pager | grep -i "no space\|disk full\|enospc"
```

Utilisez l'EC2 AWS CLI pour obtenir la sortie de console d'une instance gérée Amazon ECS

Utilisez l'ID d'instance Amazon EC2 pour récupérer la sortie de la console.

Remplacez le *user-input* avec tes valeurs.

```
aws ec2 get-console-output --instance-id instance-id --latest --output text
```

Nettoyage

Exécutez ce qui suit pour arrêter la tâche de débogage et désenregistrer la définition de tâche.

```
# Stop debug task
aws ecs stop-task --cluster $ECS_CLUSTER_NAME --task $TASK_ARN

# Deregister task definition (optional)
aws ecs deregister-task-definition --task-definition node-debugger
```

Ressources supplémentaires

Pour plus d'informations sur la résolution des problèmes liés aux instances gérées Amazon ECS, consultez les ressources suivantes :

- the section called “Erreurs liées aux instances gérées Amazon ECS”
- Résolution des problèmes
- the section called “Configuration de l'agent de conteneur”
- the section called “Surveillance des conteneurs Amazon ECS avec ECS Exec”

Résolution des problèmes liés aux instances gérées Amazon ECS

Lors du lancement de tâches avec des instances gérées Amazon ECS, Amazon ECS tente d’abord de placer les tâches sur la capacité existante et demande une capacité supplémentaire pour

les tâches qui ne peuvent pas être placées. Si l'allocation de l'instance échoue, l'ID de requête Amazon EC2 est inclus dans le message de défaillance de la tâche. Vous pouvez utiliser cet ID de demande pour consulter les détails de la demande qui a échoué dans CloudTrail pour un dépannage plus approfondi.

> **Note**
>
> Si vous choisissez d'appliquer les autorisations de moindre privilège et de spécifier vos propres autorisations pour le profil d'instance au lieu d'utiliser le `AmazonECSInstanceRolePolicyForManagedInstances` politique gérée, vous pouvez ajouter les autorisations suivantes pour vous aider à résoudre les problèmes liés aux tâches avec les instances gérées Amazon ECS :
>
> - `ecs:StartTelemetrySession`
> - `ecs:PutSystemLogEvents`

Définition des tâches incompatible avec les instances gérées Amazon ECS

Cause courante

Cette erreur se produit lorsque votre définition de tâche contient des paramètres ou des configurations qui ne sont pas pris en charge par les instances gérées Amazon ECS. Les incompatibilités courantes incluent des modes réseau, des rôles de tâches ou des besoins en ressources non pris en charge.

Résolution

1. Vérifiez que votre définition de tâche utilise `requiresCompatibilities` défini sur `MANAGED_INSTANCES`.
2. Assurez-vous que votre définition de tâche utilise le mode réseau `awsvpc`.
3. Vérifiez que les valeurs d'UC et de mémoire se situent dans les plages prises en charge pour les instances gérées Amazon ECS.
4. Consultez le message d'erreur détaillé pour obtenir des informations spécifiques sur les incompatibilités.

Fournisseur de capacité non associé au cluster

Cause courante

Cette erreur se produit lorsque le fournisseur de capacité spécifié dans votre stratégie de fournisseur de capacité n'est pas associé au cluster ou n'existe pas.

Résolution

1. Vérifiez que le fournisseur de capacité existe dans votre compte et dans votre région.
2. Associez le fournisseur de capacité à votre cluster à l'aide de la console ou de la CLI Amazon ECS.
3. Assurez-vous que le statut du fournisseur de capacité est `ACTIVE` avant de l'utiliser.

Erreurs d'autorisation liées aux rôles d'infrastructure

Cause courante

Cette erreur se produit lorsque le rôle d'infrastructure Amazon ECS ne dispose pas des autorisations nécessaires pour effectuer des opérations Amazon EC2 en votre nom, ou lorsque le rôle ne peut pas être assumé en raison de problèmes de relation de confiance.

Résolution

1. Vérifiez que votre rôle d'infrastructure entretient une relation de confiance appropriée avec Amazon ECS.
2. Assurez-vous que le rôle dispose des autorisations Amazon EC2 requises, notamment`ec2:RunInstances`, `ec2:DescribeInstances` et. `iam:PassRole`
3. Vérifiez le message d'échec d'autorisation codé dans CloudTrail pour obtenir des informations d'autorisation spécifiques.
4. Mettez à jour la politique de rôle pour inclure les autorisations manquantes identifiées dans le message d'erreur.

VcpuLimitExceeded erreur

Cause courante

Cette erreur se produit lorsque vous avez atteint votre quota de service vCPU pour la famille de types d'instances dans la région actuelle. Les instances gérées Amazon ECS ne peuvent pas lancer d'instances supplémentaires tant que la capacité n'est pas disponible.

Résolution

1. Demandez une augmentation du quota de service pour la famille de types d'instances concernée via le AWS Centre d'assistance.
2. Envisagez d'utiliser différents types d'instances relevant d'une catégorie de quotas de vCPU différente.
3. Mettez fin aux instances Amazon EC2 inutilisées pour libérer de la capacité vCPU.
4. Passez en revue la configuration de votre fournisseur de capacité afin d'utiliser des types d'instances nécessitant moins de vCPU.

InsufficientCapacity et les erreurs de capacité associées

Cause courante

Ces erreurs se produisent lorsque AWS n'a pas la capacité suffisante pour répondre à votre demande d'instance. Cela peut inclure une capacité d'instance, une capacité d'adresse ou une capacité de volume insuffisantes dans la zone de disponibilité demandée.

Résolution

1. Essayez de lancer des instances dans différentes zones de disponibilité en configurant plusieurs sous-réseaux dans votre fournisseur de capacité.
2. Envisagez d'utiliser différents types d'instances susceptibles de disposer d'une plus grande capacité disponible.
3. Attendez et recommencez l'opération, car la disponibilité des capacités change fréquemment.
4. Pour des besoins de capacité persistants, pensez à utiliser des instances réservées ou des Savings Plans.

UnauthorizedOperation erreur

Cause courante

Cette erreur se produit lorsque le service Amazon ECS ne dispose pas des autorisations nécessaires pour effectuer des opérations Amazon EC2 ou transmettre des rôles IAM. Les scénarios courants incluent des autorisations `ec2:RunInstances` manquantes ou des autorisations `iam:PassRole` pour le profil d'instance.

Résolution

1. Vérifiez que votre rôle d'infrastructure Amazon ECS dispose des autorisations nécessaires pour lancer des instances Amazon EC2.
2. Assurez-vous que le rôle d'infrastructure dispose d'autorisations `iam:PassRole` pour le profil d'instance utilisé par vos instances gérées Amazon ECS.
3. Vérifiez le message d'échec d'autorisation codé dans CloudTrail pour obtenir des informations d'autorisation spécifiques.
4. Mettez à jour la politique de rôle pour inclure les autorisations manquantes identifiées dans le message d'erreur.

La tâche a expiré pendant l'attente de la capacité.

Cause courante

Cette erreur se produit lorsque le lancement et l'enregistrement des instances auprès du cluster prennent plus de temps que prévu. Cela peut être dû à des contraintes de capacité Amazon EC2, à des échecs de lancement d'instance ou à des problèmes de connectivité réseau.

Résolution

1. Vérifiez l'état du service Amazon EC2 dans votre région pour tout problème persistant.
2. Vérifiez que vos sous-réseaux disposent de suffisamment d'adresses IP disponibles.
3. Assurez-vous que vos groupes de sécurité autorisent le trafic nécessaire à la communication avec les agents Amazon ECS.
4. Envisagez d'utiliser plusieurs zones de disponibilité pour améliorer la disponibilité des capacités.
5. Réessayez l'opération de lancement de la tâche, car les contraintes de capacité sont souvent temporaires.

Erreurs de configuration réseau

Cause courante

Ces erreurs se produisent lorsqu'il existe des incohérences entre les exigences réseau de votre tâche et la configuration réseau du fournisseur de capacité, telles que des incompatibilités VPC ou une configuration réseau manquante.

Résolution

1. Vérifiez que votre fournisseur de capacité est configuré avec le VPC et les sous-réseaux appropriés.
2. Assurez-vous que les groupes de sécurité et les sous-réseaux appartiennent au même VPC.
3. Vérifiez que la configuration réseau de votre définition de tâche est compatible avec le fournisseur de capacité.
4. Mettez à jour la configuration de votre fournisseur de capacité avec les paramètres réseau appropriés.

Le fournisseur de capacité ne peut pas être supprimé en raison d'instances bloquées

Cause courante

Ces erreurs se produisent lorsque les instances gérées Amazon ECS sont bloquées dans un `ACTIVE` ou `DRAINING` état mais aucune tâche n'est en cours d'exécution sur les instances.

Résolution

Pour autoriser la suppression du fournisseur de capacité, vous pouvez forcer le désenregistrement des instances bloquées à l'aide de la commande suivante.

```
aws ecs deregister-container-instance \
    --cluster arn:aws:ecs:us-east-1:111122223333:cluster/MyCluster \
    --container-instance arn:aws:ecs:us-east-1:111122223333:container-instance/
a1b2c3d4-5678-90ab-cdef-11111EXAMPLE \
    --force
```

Résolution des erreurs liées aux instances gérées Amazon ECS

Vous trouverez ci-dessous quelques messages d'erreur relatifs aux instances gérées Amazon ECS et les mesures que vous pouvez prendre pour corriger ces erreurs.

Le fournisseur d’instances gérées Amazon ECS n’est pas pris en charge sans nom de cluster

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité avec un paramètre de fournisseur d’instances gérées Amazon ECS, mais aucun champ de cluster.

Pour résoudre ce problème, spécifiez un nom de cluster valide lors de la création du fournisseur de capacité.

Les informations relatives au fournisseur d’instances gérées Amazon ECS sont requises lors de la création d’un fournisseur de capacité d’instances gérées Amazon ECS.

Cette erreur apparaît lorsque vous essayez de créer un fournisseur de capacité avec le fournisseur d’instances gérées Amazon ECS, mais que vous ne fournissez pas l’objet réel.

Pour résoudre ce problème, spécifiez des informations valides sur le fournisseur d’instances gérées Amazon ECS et réessayez.

Le fournisseur de capacité des instances gérées Amazon ECS doit spécifier un profil d’instance

Cette erreur s'affiche lorsque vous tentez de créer un fournisseur de capacité avec le fournisseur Amazon ECS Managed Instances mais que vous ne fournissez pas Ec2InstanceProfile.

Pour résoudre ce problème, spécifiez un profil d’instance EC2 valide pour votre fournisseur de capacité d’instances gérées Amazon ECS. Pour plus d’informations, consultez the section called “Profil d’instance des instances gérées Amazon ECS”.

Le fournisseur de capacité Amazon ECS Managed Instances doit spécifier InfrastructureRole

Ce message s'affiche lorsque vous essayez de créer un fournisseur de capacité avec le fournisseur Amazon ECS Managed Instances mais que vous ne fournissez pas InfrastructureRole.

Pour corriger cela, spécifiez un rôle d’infrastructure valide pour votre fournisseur de capacité d’instances gérées Amazon ECS. Pour plus d’informations, consultez the section called “Rôle IAM d’infrastructure”.

Le fournisseur de capacité des instances gérées Amazon ECS doit spécifier une configuration réseau

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité avec le fournisseur Amazon ECS Managed Instances sans fournir de configuration réseau.

Pour résoudre ce problème, spécifiez une configuration réseau valide avec des sous-réseaux non vides pour votre fournisseur de capacité d'instances gérées Amazon ECS. Pour plus d'informations, consultez the section called "Mise en réseau des tâches pour les instances gérées Amazon ECS".

Aucun type d'instance ne répond aux exigences d'instance spécifiées dans le fournisseur de capacité d'instances gérées Amazon ECS

Cela se produit lorsque vous essayez de créer un fournisseur de capacité avec le fournisseur d'instances gérées Amazon ECS, mais qu'aucun type d'instance EC2 ne répond aux exigences d'instance.

Pour résoudre ce problème, passez en revue et ajustez les exigences de vos instances pour qu'elles correspondent aux types d'instances EC2 disponibles.

L'ARN du rôle d'infrastructure spécifié n'est pas valide

Cette erreur se produit lorsque InfrastructureRole ne suit pas le format ARN spécifique.

Format attendu : `arn:partition:iam::account-id:role/role-name`. Spécifiez un ARN de rôle valide et réessayez. Pour plus d'informations, consultez the section called "Rôle IAM d'infrastructure".

L'ARN du rôle de profil d'instance spécifié n'est pas valide.

Cette erreur se produit lorsque le profil d'instance ne suit pas le format ARN spécifique.

Format attendu : `arn:partition:iam::account-id:instance-profile/profile-name`. Spécifiez un ARN de rôle valide et réessayez. Pour plus d'informations, consultez the section called "Profil d'instance des instances gérées Amazon ECS".

L'identifiant du groupe de sécurité spécifié n'est pas valide

Cette erreur se produit lorsque le groupe de sécurité ne suit pas le format ARN spécifique.

Format attendu : `sg-xxxxxxxx` ou `sg-xxxxxxxxxxxxxxxxx` (8 ou 17 caractères, y compris des lettres (minuscules) et des chiffres après « subnet- »). Spécifiez un ID de groupe de sécurité et réessayez.

L'identifiant de sous-réseau spécifié n'est pas valide

Cette erreur se produit lorsque l'ID de sous-réseau ne suit pas le format spécifique.

Format attendu : `subnet-xxxxxxxx` ou `subnet-xxxxxxxxxxxxxxxxx` (8 ou 17 caractères, y compris des lettres (minuscules) et des chiffres après « subnet- »). Spécifiez un ID de sous-réseau et réessayez.

Le fournisseur de capacité des instances gérées Amazon ECS doit spécifier une configuration réseau avec des sous-réseaux non vides

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité avec le fournisseur d'instances gérées Amazon ECS et une configuration réseau avec des sous-réseaux vides.

Pour résoudre ce problème, spécifiez une configuration réseau valide avec des sous-réseaux non vides pour votre fournisseur de capacité d'instances gérées Amazon ECS.

Le fournisseur de capacité des instances gérées Amazon ECS ne peut pas avoir un nombre de balises supérieur à la valeur maximale autorisée

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité avec un nombre de balises supérieur au maximum autorisé (45).

Pour résoudre ce problème, réduisez le nombre de balises à 45 ou moins et réessayez.

Valeur non valide pour les propagateTags

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité avec une valeur propagateTags incorrecte.

La valeur doit être l'une des valeurs propagateTags valides. Spécifiez une valeur valide et réessayez.

Le fournisseur de capacité spécifié existe déjà au niveau de l'étendue du cluster

Cette erreur se produit lorsque vous essayez de créer, de mettre à jour ou de supprimer un fournisseur de capacité d'instances gérées Amazon ECS sans nom de cluster, mais qu'il existe un fournisseur de capacité au niveau du cluster.

Pour modifier la configuration ou supprimer le fournisseur de capacité, mettez à jour ou supprimez le fournisseur de capacité avec le paramètre de cluster.

Le fournisseur de capacité spécifié existe déjà au niveau du compte ou d'un autre cluster

Cette erreur se produit lorsque vous essayez de créer, de mettre à jour ou de supprimer un fournisseur de capacité d'instances gérées Amazon ECS avec un champ cluster, alors qu'il existe déjà un fournisseur de capacité au niveau du compte ou un fournisseur de capacité au niveau du cluster pour un autre cluster.

Pour résoudre ce problème, utilisez un autre nom de fournisseur de capacité ou utilisez le fournisseur de capacité existant.

Le nom du cluster actif du fournisseur de capacité ne correspond pas au nom du cluster indiqué dans la requête

Cette erreur se produit lorsque le nom du cluster actif du fournisseur de capacité ne correspond pas au nom du cluster spécifié dans la requête.

Assurez-vous que le nom du cluster indiqué dans votre requête correspond au cluster associé au fournisseur de capacité.

Le fournisseur de capacité n'est pas le fournisseur de capacité des instances gérées Amazon ECS

Cette erreur se produit lorsque vous essayez d'utiliser un fournisseur de capacité qui n'est pas un fournisseur de capacité d'instances gérées Amazon ECS dans un contexte qui en nécessite un.

Assurez-vous d'utiliser un fournisseur de capacité d'instances gérées Amazon ECS valide pour votre requête.

CapacityProvider le nom ne doit pas être vide

Cette erreur se produit lorsque vous essayez de créer un fournisseur de capacité sans spécifier de nom de fournisseur de capacité.

Spécifiez un nom de fournisseur de capacité valide et réessayez.

Un nom est requis lors de la mise à jour d'un fournisseur de capacité

Cette erreur se produit lorsque vous essayez de mettre à jour un fournisseur de capacité sans spécifier de nom de fournisseur de capacité.

Spécifiez un nom valide et réessayez.

Les balises ne peuvent pas être nulles ou comporter des éléments nuls

Cette erreur se produit lorsque vous essayez d'étiqueter un fournisseur de capacité avec des valeurs nulles.

Assurez-vous que toutes les valeurs des balises sont correctement spécifiées et qu'elles ne sont pas nulles.

Motifs d'échec d'API Amazon ECS

Si une action API que vous avez déclenchée via l'API Amazon ECS, la console ou la AWS CLI renvoie un message d'erreur `failures`, les éléments suivants peuvent aider à résoudre la cause. L'échec renvoie un motif et l'Amazon Resource Name (ARN) de la ressource associée à l'échec.

De nombreuses ressources sont Region-specific, donc lorsque vous utilisez la console, assurez-vous de définir la région appropriée pour vos ressources. Lors de l'utilisation du AWS CLI, assurez-vous que votre AWS CLI les commandes sont envoyées à la bonne région avec le `--region` *`region`* paramètre.

Pour plus d'informations sur la structure du type de données `Failure`, consultez Failure (Échec) dans la Référence d'API Amazon Elastic Container Service.

Voici des exemples de messages d'échec que vous pouvez recevoir lors de l'exécution de commandes API.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`DescribeClusters`	`MISSING`	Le cluster spécifié n'a pas été trouvé. Vérifiez l'orthographe du nom du cluster.
`DescribeInstances`	`MISSING`	L'instance de conteneur spécifiée n'a pas été trouvée.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
		Vérifiez que vous avez spécifié le cluster dans lequel l'instance de conteneur est enregistrée et que l'ARN ou l'ID de l'instance de conteneur est correct.
`DescribeServices`	`MISSING`	Le service spécifié n'a pas été trouvé. Vérifiez que la région ou le cluster correct est spécifié et que l'ARN ou le nom du service est valide.
`DescribeTasks`	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que la région ou le cluster correct est spécifié et que l'ID ou l'ARN de la tâche est valide.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
`DescribeTasks`	`TaskFailedToStart: RESOURCE:*`	En cas d'erreur `RESOURCE: CPU` , le nombre de processeurs demandé par la tâche n'est pas disponible sur vos instances de conteneur. Cela se produit généralement lorsque l’UC requise dans la définition de votre tâche est supérieure à la taille de l’UC des instances EC2 définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité. En cas d'erreur `RESOURCE: MEMORY` , la quantité de mémoire demandée par la tâche n'est pas disponible sur vos instances de conteneur . Cela se produit généralement lorsque la quantité de mémoire requise dans la définition de votre tâche est supérieure à la mémoire prise en charge des instances EC2 définies dans le groupe Auto Scaling mappé au fournisseur de capacité. Vous devez vérifier la configuration de votre fournisseur de capacité.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: AGENT`	L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellem ent déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée. Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.
	`TaskFailedToStart: MemberOf placement constraint unsatisfi ed`	Aucune instance de conteneur ne répond aux contraintes de placement définies dans votre définition de tâche.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: ATTRIBUTE`	La définition de votre tâche contient un paramètre nécessitant un attribut d'instance de conteneur spécifique qui n'est pas disponible sur vos instances de conteneur. Cela se produit, par exemple, si votre tâche utilise le mode réseau `awsvpc`, mais qu'aucune instance dans les sous-réseaux spécifiés ne comporte l'attribut `ecs.capability.task-eni` . Pour plus d'informations sur les attributs requis pour des paramètres de définition de tâche et des variables de configuration d'agent spécifiques, consultez Paramètres de définition de tâche Amazon ECS pour Fargate et Configuration de l'agent de conteneur Amazon ECS.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`TaskFailedToStart: NO ACTIVE INSTANCES`	Il n'y a aucune instance active dans votre fournisseur de capacité. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, veuillez consulter Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling (langue française non garantie).
	`TaskFailedToStart: EMPTY CAPACITY PROVIDER`	Il n'existe pas d'instances dans votre cluster. Cela est probablement dû à un fournisseur de capacité vide ou au fait que les instances du fournisseur de capacité ne sont pas enregistrées auprès du cluster. Pour plus d'informations sur la gestion de vos groupes Auto Scaling, veuillez consulter Groupes Auto Scaling dans le Guide de l'utilisateur Amazon EC2 Auto Scaling (langue française non garantie).
`GetTaskProtection`	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que l'ARN ou le nom du cluster et l'ARN ou l'ID de tâche sont valides.

Action d’API	Motif de l'échec ou de l'arrêt	Cause
	TASK_NOT_VALID	La tâche spécifiée ne fait pas partie d’un service Amazon ECS. Seules les tâches gérées par les services Amazon ECS peuvent être protégées. Vérifiez l'ARN ou l'ID de la tâche et réessayez.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`RunTask` ou `StartTask`	`RESOURCE:*`	La ou les ressources demandées par la tâche ne sont pas disponibles sur les instances de conteneur du cluster. Si la ressource demandée concerne le CPU, la mémoire, les ports ou les interfaces réseau Elastic, vous devrez peut-être ajouter d'autres instances de conteneur à votre cluster. Les erreurs `RESOURCE: ENI` indiquent que votre cluster ne dispose pas de points d'attache d'interface réseau Elastic, qui sont requis pour les tâches utilisant le mode réseau `awsvpc`. Le nombre d'interfaces réseau qui peuvent être attachées à des instances Amazon EC2 est limité et l'interface réseau principale est considérée comme l'une d'elles. Pour en savoir plus sur le nombre d'interfaces réseau prises en charge par type d'instance, consultez la section Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
		Pour `RESOURCE:GPU` erreurs, le nombre de GPU requis par la tâche n'est pas disponible et vous devrez peut-être ajouter GPU-enabl ed instances de conteneurs vers votre cluster. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.
	`AGENT`	L'instance de conteneur avec laquelle vous avez essayé de lancer une tâche comporte un agent qui est actuellem ent déconnecté. Afin d'éviter de longs délais d'attente pour le placement de la tâche, la demande a été rejetée. Pour plus d'informations sur la résolution problèmes des agents déconnectés, consultez Comment résoudre les problèmes liés à un agent Amazon ECS déconnecté ?.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
	`LOCATION`	L'instance de conteneur sur laquelle vous avez tenté de lancer une tâche se trouve dans une zone de disponibilité différente de celle du ou des sous-réseaux que vous avez spécifiés dans votre `awsVpcConfiguration` .
	`ATTRIBUTE`	La définition de votre tâche contient un paramètre nécessitant un attribut d'instance de conteneur spécifique qui n'est pas disponible sur vos instances de conteneur. Cela se produit, par exemple, si votre tâche utilise le mode réseau `awsvpc`, mais qu'aucune instance dans les sous-réseaux spécifiés ne comporte l'attribut `ecs.capability.task-eni` . Pour plus d'informations sur les attributs requis pour des paramètres de définition de tâche et des variables de configuration d'agent spécifiques, consultez Paramètres de définition de tâche Amazon ECS pour Fargate et Configuration de l'agent de conteneur Amazon ECS.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`StartTask`	`MISSING`	L'instance de conteneur sur laquelle vous avez tenté de lancer la tâche est introuvable. Vérifiez si le cluster ou la région spécifiés sont incorrects, ou si l'ARN ou l'ID de l'instance de conteneur est mal orthographié.
	`INACTIVE`	L'enregistrement de l'instance de conteneur sur laquelle vous avez essayé de lancer une tâche a été précédemment annulé avec Amazon ECS et cette instance ne peut pas être utilisée.
`StopServiceDeployment`	`ECS deployment failed`	Un compte frauduleux a utilisé l'API `StopServiceDeployment` .
`TagResource`	`InvalidParameterException`	L'ARN du service que vous étiquetez est au format court. Vous devez passer au format long. Pour plus d'informations sur le processus de migration de l'ARN, consultez la section Migration d'un ARN court de service Amazon ECS vers un ARN long.

Action d'API	Motif de l'échec ou de l'arrêt	Cause
`UpdateTaskProtection`	`DEPLOYMENT_BLOCKED`	Impossible de définir la protection des tâches, car une ou plusieurs tâches protégées empêchent le déploiement de service d'atteindre un état stable. Désactivez la protection des tâches sur les tâches existantes ou attendez que la protection des tâches expire.
	`MISSING`	La tâche spécifiée n'a pas été trouvée. Vérifiez que l'ARN ou le nom du cluster et l'ARN ou l'ID de tâche sont valides.
	`TASK_NOT_VALID`	La tâche spécifiée ne fait pas partie d'un service Amazon ECS. Seules les tâches gérées par les services Amazon ECS peuvent être protégées. Vérifiez l'ARN ou l'ID de la tâche et réessayez.

> **Note**
>
> Outre les scénarios d'échecs décrits ici, les opérations d'API peuvent également échouer en raison d'exceptions, ce qui entraîne des réponses d'erreur. Pour obtenir la liste de ces exceptions, consultez la section Erreurs courantes.

Résolution des problèmes avec Amazon Q Developer

Vous pouvez utiliser Développeur Amazon Q dans la console Amazon ECS pour vous aider à diagnostiquer et à résoudre les problèmes liés à vos ressources Amazon ECS. Pour certaines erreurs

et certains messages d'état relatifs aux conteneurs, aux tâches, aux services, aux déploiements et aux définitions de tâches, la console affiche un Inspectez avec Amazon Q Developer option. Lorsque vous choisissez cette option, Amazon Q Developer analyse le problème dans son contexte et suggère des causes possibles ainsi que des mesures correctives.

Autorisations requises

- Autorisations pour afficher les ressources Amazon ECS que vous souhaitez résoudre, telles que les clusters, les services, les tâches et les définitions de tâches.
- Autorisations d'utilisation d'Amazon Q Developer dans la console.
- Autorisations (recommandées) pour consulter les journaux et les mesures associés, tels que :
  - CloudWatch Journaux
  - CloudWatch

Procédure

1. Ouvrez la console à https://console.aws.amazon.com/ecs/v2.
2. Déterminez la ressource que vous souhaitez dépanner.

Ressource	Étapes
Tâches	1. Sur la page Clusters, choisissez le cluster. 2. Choisissez l'onglet Tasks. 3. Choisissez la tâche que vous souhaitez étudier.
Containers	1. Sur la page Clusters, choisissez le cluster. 2. Choisissez le Tâches onglet, puis choisissez la tâche. 3. Dans le Conteneurs section, choisissez le conteneur que vous souhaitez examiner.
Services et déploiements	1. Sur la page Clusters, choisissez le cluster. 2. Choisissez l'onglet Services.

Ressource	Étapes
	3. Choisissez le service, puis passez en revue le Déploiements section.
Définitions de tâche	1. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches). 2. Choisissez la famille de définitions de tâches, puis choisissez la révision que vous souhaitez étudier.

3. Sur la page des détails de la ressource, recherchez l'état ou la raison de santé qui décrit le problème.
4. Cliquez sur la raison du statut pour ouvrir la fenêtre contextuelle.
5. Si disponible, choisissez Inspectez avec Amazon Q Developer.
6. Consultez l'explication et les étapes de correction suggérées par Amazon Q Developer. Appliquez les modifications de configuration ou de fonctionnement appropriées à votre environnement.

Considérations

Lorsque vous utilisez Amazon Q Developer avec Amazon ECS, tenez compte des points suivants :

- Disponibilité des boutons - Le bouton « Inspecter avec Amazon Q Developer » s'affiche uniquement pour les ressources susceptibles de rencontrer des problèmes. Cette option n'est pas disponible pour les ressources saines.
- Read-only opérations - L'intégration Amazon Q Developer effectue uniquement des opérations de lecture. Il n'effectue aucune action de mutation ou d'écriture.
- Cross-region En traitement - Amazon Q Developer peut traiter des données entre AWS régions à fournir AI-powered analyse. Pour plus d'informations sur le traitement interrégional, voir Cross-region traitement dans Amazon Q Developer.
- Console uniquement - Cette intégration est disponible uniquement via la console. Il n'est pas disponible via le AWS CLI, AWS API, ou infrastructure en tant qu'outils de code.

En savoir plus

Pour plus d'informations sur l'utilisation d'Amazon Q Developer, consultez Discuter avec un développeur Amazon Q à propos de AWS.

Sécurité dans Amazon Elastic Container Service

La sécurité du cloud chez AWS est la plus haute priorité. En tant que AWS client, vous bénéficiez d'un centre de données et d'une architecture réseau conçus pour répondre aux exigences des organisations les plus sensibles en matière de sécurité.

La sécurité est une responsabilité partagée entre AWS et toi. Le modèle de responsabilité partagée décrit cette notion par les termes sécurité du cloud et sécurité dans le cloud :

- Sécurité du cloud — AWS est responsable de la protection de l'infrastructure qui fonctionne AWS services dans le AWS Nuage. AWS vous fournit également des services que vous pouvez utiliser en toute sécurité. Third-party les auditeurs testent et vérifient régulièrement l'efficacité de notre sécurité dans le cadre du AWS programmes de conformité. Pour en savoir plus sur les programmes de conformité qui s'appliquent à Amazon Elastic Container Service, veuillez consulter Services AWS concernés par le programme de conformité.
- Sécurité dans le cloud — Votre responsabilité est déterminée par le AWS service que vous utilisez. Vous êtes également responsable d'autres facteurs, y compris de la sensibilité de vos données, des exigences de votre entreprise, ainsi que de la législation et de la réglementation applicables.

Cette documentation vous aide à comprendre comment appliquer le modèle de responsabilité partagée lors de l'utilisation d'Amazon ECS. Les rubriques suivantes vous montrent comment configurer Amazon ECS pour répondre à vos objectifs de sécurité et de conformité. Vous apprenez également à utiliser d'autres AWS des services qui vous aident à surveiller et à sécuriser vos ressources Amazon ECS.

Rubriques

- Gestion des identités et des accès pour Amazon Elastic Container Service
- Journalisation et surveillance dans Amazon Elastic Container Service
- Validation de la conformité pour Amazon Elastic Container Service
- AWS Fargate Norme fédérale de traitement de l'information (FIPS-140)
- Sécurité de l'infrastructure dans Amazon Elastic Container Service
- AWS modèle de responsabilité partagée pour Amazon ECS
- Modèle de responsabilité partagée pour les instances gérées Amazon ECS
- Pratiques exemplaires en matière de sécurité réseau pour Amazon ECS
- Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS

Gestion des identités et des accès pour Amazon Elastic Container Service

Gestion des identités et des accès AWS (IAM) est un Service AWS qui permet à un administrateur de contrôler en toute sécurité l'accès à AWS ressources. Des administrateurs IAM contrôlent les personnes qui peuvent être authentifiées (connectées) et autorisées (disposant d'autorisations) à utiliser des ressources Amazon ECS. IAM est un Service AWS que vous pouvez utiliser sans frais supplémentaires.

Rubriques

- Public ciblé
- Authentification par des identités
- Gestion de l’accès à l’aide de politiques
- Fonctionnement d'Amazon Elastic Container Service avec IAM
- Identity-based exemples de politiques pour Amazon Elastic Container Service
- AWS politiques gérées pour Amazon Elastic Container Service
- Utilisation des rôles liés à un service pour Amazon ECS
- Rôles IAM pour Amazon ECS
- Autorisations requises pour la console Amazon ECS
- Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements
- Autorisations IAM requises pour l’autoscaling d’un service Amazon ECS
- Octroi de l'autorisation de baliser les ressources lors de la création
- Dépannage de l'identité et de l'accès Amazon Elastic Container Service
- Pratiques exemplaires relatives à IAM pour Amazon ECS

Public ciblé

Comment utilisez-vous Gestion des identités et des accès AWS (IAM) diffère en fonction de votre rôle :

- Utilisateur du service : demandez des autorisations à votre administrateur si vous ne pouvez pas accéder aux fonctionnalités (voir Dépannage de l'identité et de l'accès Amazon Elastic Container Service)

- Administrateur du service : déterminez l’accès des utilisateurs et soumettez les demandes d’autorisation (voir Fonctionnement d'Amazon Elastic Container Service avec IAM)
- Administrateur IAM : rédigez des politiques pour gérer l’accès (voir Identity-based exemples de politiques pour Amazon Elastic Container Service)

Authentification par des identités

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

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), authentification unique, ou Google/Facebook informations d'identification. Pour plus d’informations sur la connexion, consultez Connexion à votre Compte AWS dans le Guide de l’utilisateur Connexion à AWS .

Pour un accès programmatique, 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.

Compte AWS utilisateur root

Lorsque vous créez un Compte AWS, vous commencez avec une seule identité de connexion appelée Compte AWS utilisateur root qui a un accès complet à tous Services AWS et 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.

Identité fédérée

Il est recommandé d'obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder Services AWS en utilisant des informations d'identification temporaires.

UNE identité fédérée est un utilisateur de l'annuaire de votre entreprise, de votre fournisseur d'identité Web ou Directory Service qui accède Services AWS en utilisant des informations d'identification provenant d'une source d'identité. Les identités fédérées assument des rôles qui fournissent des informations d’identification temporaires.

Pour une gestion des accès centralisée, nous vous recommandons d'utiliser AWS IAM Identity Center. Pour plus d'informations, consultez Qu'est-ce que IAM Identity Center ? dans le Guide de l'utilisateur AWS IAM Identity Center .

Utilisateurs et groupes 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 Obliger les utilisateurs humains à utiliser la fédération avec un fournisseur d'identité pour accéder AWS en utilisant des 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.

Rôles IAM

Un rôle IAM est une identité dotée d'autorisations spécifiques qui fournit des informations d'identification temporaires. Vous pouvez assumer un rôle en passage d'un rôle utilisateur à un rôle IAM (console) ou en appelant un AWS CLI ou AWS Fonctionnement de l'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.

Gestion de l'accès à l'aide de politiques

Vous contrôlez l'accès dans AWS en créant des politiques et en les rattachant à AWS identités ou ressources. Une politique définit les autorisations lorsqu'elles sont associées à une identité ou à une ressource. AWS évalue ces politiques lorsqu'un directeur fait une demande. La plupart des politiques sont stockées dans AWS sous forme de documents JSON. Pour plus d'informations les documents de politique JSON, consultez Vue d'ensemble des politiques JSON dans le Guide de l'utilisateur IAM.

À l'aide de politiques, les administrateurs précisent qui a accès à quoi en définissant quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

Par défaut, les utilisateurs et les rôles ne disposent d'aucune autorisation. Un administrateur IAM crée des politiques IAM et les ajoute aux rôles, que les utilisateurs peuvent ensuite assumer. Les politiques IAM définissent les autorisations quelle que soit la méthode que vous utilisez pour exécuter l'opération.

Identity-based politiques

Identity-based les politiques sont des documents de politique d'autorisations JSON que vous associez à une identité (utilisateur, groupe ou rôle). Ces politiques contrôlent les actions que peuvent exécuter ces identités, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l'identité, consultez Définition d'autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l'utilisateur IAM.

Identity-based les politiques peuvent être politiques en ligne (intégré directement dans une identité unique) ou politiques gérées (politiques autonomes associées à plusieurs identités). Pour découvrir comment choisir entre des politiques gérées et en ligne, consultez Choix entre les politiques gérées et les politiques en ligne dans le Guide de l'utilisateur IAM.

Resource-based politiques

Resource-based les politiques sont des documents de stratégie JSON que vous joignez à une ressource. Les exemples incluent les politiques de confiance de rôle IAM et les stratégies de compartiment Amazon S3. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Vous devez spécifier un principal dans une politique basée sur les ressources.

Resource-based les politiques sont des politiques en ligne situées dans ce service. Vous ne pouvez pas utiliser AWS politiques gérées depuis IAM dans le cadre d'une politique basée sur les ressources.

Autres types de politique

AWS prend en charge des types de politiques supplémentaires qui peuvent définir les autorisations maximales accordées par les types de politiques les plus courants :

- Limites d'autorisations : une limite des autorisations définit le nombre maximum d'autorisations qu'une politique basée sur l'identité peut accorder à une entité IAM. Pour plus d'informations, consultez Limites d'autorisations pour des entités IAM dans le Guide de l'utilisateur IAM.

- Politiques de contrôle des services (SCP) : spécifient les autorisations maximales pour une organisation ou une unité organisationnelle dans AWS Organizations. Pour plus d'informations, consultez Politiques de contrôle de service dans le Guide de l'utilisateur AWS Organizations .
- Politiques de contrôle des ressources (RCP) : définissent les autorisations maximales disponibles pour les ressources de votre organisation. Pour plus d'informations, consultez Politiques de contrôle des ressources (RCP) dans le Guide de l'utilisateur AWS Organizations .
- Politiques de session : politiques avancées que vous passez en tant que paramètre lorsque vous créez par programmation une session temporaire pour un rôle ou un utilisateur fédéré. Pour plus d'informations, consultez Politiques de session dans le Guide de l'utilisateur IAM.

Plusieurs types de politique

Lorsque plusieurs types de politiques s'appliquent à la requête, les autorisations en résultant sont plus compliquées à comprendre. Pour savoir comment AWS détermine s'il faut autoriser une demande lorsque plusieurs types de politiques sont impliqués, voir Logique d'évaluation des politiques dans le Guide de l'utilisateur IAM.

Fonctionnement d'Amazon Elastic Container Service avec IAM

Avant d'utiliser IAM pour gérer l'accès à Amazon ECS, apprenez les fonctions IAM qui peuvent être utilisées dans cette situation.

Fonctionnalité IAM	Prise en charge d'Amazon ECS
Identity-based politiques	Oui
Resource-based politiques	Non
Actions de politique	Oui
Ressources de politique	Partielle
Clés de condition de politique	Oui
ACL	Non
ABAC (étiquettes dans les politiques)	Oui

Fonctionnalité IAM	Prise en charge d'Amazon ECS
Informations d’identification temporaires	Oui
Transmission des sessions d’accès (FAS)	Oui
Rôles de service	Oui
Service-linked rôles	Oui

Pour obtenir une vue d'ensemble de la façon dont Amazon ECS et d'autres AWS les services fonctionnent avec la plupart des fonctionnalités IAM, voir AWS services qui fonctionnent avec IAM dans le Guide de l'utilisateur IAM.

Identity-based politiques pour Amazon ECS

Prend en charge les politiques basées sur l’identité : oui

Identity-based les politiques sont des documents de politique d'autorisations JSON que vous pouvez joindre à une identité, telle qu'un utilisateur, un groupe d'utilisateurs ou un rôle IAM. Ces politiques contrôlent quel type d’actions des utilisateurs et des rôles peuvent exécuter, sur quelles ressources et dans quelles conditions. Pour découvrir comment créer une politique basée sur l’identité, consultez Définition d’autorisations IAM personnalisées avec des politiques gérées par le client dans le Guide de l’utilisateur IAM.

Avec les politiques IAM basées sur l’identité, vous pouvez spécifier des actions et ressources autorisées ou refusées, ainsi que les conditions dans lesquelles les actions sont autorisées ou refusées. Pour découvrir tous les éléments que vous utilisez dans une politique JSON, consultez Références des éléments de politique JSON IAM dans le Guide de l’utilisateur IAM.

Identity-based exemples de politiques pour Amazon ECS

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Identity-based exemples de politiques pour Amazon Elastic Container Service.

Resource-based politiques au sein d'Amazon ECS

Prend en charge les politiques basées sur les ressources : non

Resource-based les politiques sont des documents de stratégie JSON que vous joignez à une ressource. Par exemple, les politiques de confiance de rôle IAM et les politiques de compartiment Amazon S3 sont des politiques basées sur les ressources. Dans les services qui sont compatibles avec les politiques basées sur les ressources, les administrateurs de service peuvent les utiliser pour contrôler l'accès à une ressource spécifique. Pour la ressource dans laquelle se trouve la politique, cette dernière définit quel type d'actions un principal spécifié peut effectuer sur cette ressource et dans quelles conditions. Vous devez spécifier un principal dans une politique basée sur les ressources. Les principaux peuvent inclure des comptes, des utilisateurs, des rôles, des utilisateurs fédérés ou Services AWS.

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. Pour plus d'informations, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Actions de politique pour Amazon ECS

Prend en charge les actions de politique : oui

Les administrateurs peuvent utiliser AWS Politiques JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Action` d'une politique JSON décrit les actions que vous pouvez utiliser pour autoriser ou refuser l'accès à une politique. Intégration d'actions dans une politique afin d'accorder l'autorisation d'exécuter les opérations associées.

Pour afficher la liste des actions Amazon ECS, consultez Actions définies par Amazon Elastic Container Service dans Référence de l'autorisation de service.

Les actions de politique dans Amazon ECS utilisent le préfixe suivant avant l'action :

```
ecs
```

Pour indiquer plusieurs actions dans une seule déclaration, séparez-les par des virgules.

```
"Action": [
      "ecs:action1",
      "ecs:action2"
         ]
```

Vous pouvez aussi spécifier plusieurs actions à l'aide de caractères génériques (*). Par exemple, pour spécifier toutes les actions qui commencent par le mot `Describe`, incluez l'action suivante :

```
"Action": "ecs:Describe*"
```

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Identity-based exemples de politiques pour Amazon Elastic Container Service.

Ressources de politique pour Amazon ECS

Prend en charge les ressources de politique : partiellement

Les administrateurs peuvent utiliser AWS Politiques JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément de politique JSON `Resource` indique le ou les objets auxquels l'action s'applique. Il est recommandé de définir une ressource à l'aide de son Amazon Resource Name (ARN). Pour les actions qui ne sont pas compatibles avec les autorisations de niveau ressource, utilisez un caractère générique (*) afin d'indiquer que l'instruction s'applique à toutes les ressources.

```
"Resource": "*"
```

Pour afficher la liste des types de ressources Amazon ECS et leur ARN, consultez Ressources définies par Amazon Elastic Container Service dans la Référence de l'autorisation de service. Pour connaître les actions avec lesquelles vous pouvez spécifier l'ARN de chaque ressource, consultez Actions définies par Amazon Elastic Container Service.

Certaines actions d'API Amazon ECS prennent en charge plusieurs ressources. Par exemple, plusieurs clusters peuvent être référencés lors de l'appel de l'action d'API `DescribeClusters`. Pour spécifier plusieurs ressources dans une seule instruction, séparez leurs ARN par des virgules.

```
"Resource": [
      "EXAMPLE-RESOURCE-1",
      "EXAMPLE-RESOURCE-2"
```

Par exemple, la ressource de cluster Amazon ECS a l'ARN suivant :

```
arn:${Partition}:ecs:${Region}:${Account}:cluster/${clusterName}
```

Pour spécifier le cluster `my-cluster-1` et `my-cluster-2` dans votre instruction, utilisez les ARN suivants :

```
"Resource": [
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-1",
        "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster-2"
```

Pour spécifier tous les clusters qui appartiennent à un compte spécifique, utilisez le caractère générique (*) :

```
"Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/*"
```

Pour les définitions de tâches, vous pouvez spécifier la dernière révision ou une révision spécifique.

Pour spécifier toutes les révisions de la définition de tâche, utilisez le caractère générique (*) :

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:*"
```

Pour spécifier une révision de définition de tâche spécifique, utilisez $ {TaskDefinitionRevisionNumber} :

```
"Resource:arn:${Partition}:ecs:${Region}:${Account}:task-definition/
${TaskDefinitionFamilyName}:${TaskDefinitionRevisionNumber}"
```

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Identity-based exemples de politiques pour Amazon Elastic Container Service.

Clés de condition de politique pour Amazon ECS

Prend en charge les clés de condition de politique spécifiques au service : oui

Les administrateurs peuvent utiliser AWS Politiques JSON pour spécifier qui a accès à quoi. C'est-à-dire, quel principal peut effectuer des actions sur quelles ressources et dans quelles conditions.

L'élément `Condition` indique à quel moment les instructions s'exécutent en fonction de critères définis. Vous pouvez créer des expressions conditionnelles qui utilisent des opérateurs de condition,

tels que les signes égal ou inférieur à, pour faire correspondre la condition de la politique aux valeurs de la demande. Pour tout voir AWS clés de condition globales, voir AWS clés de contexte de condition globale dans le Guide de l'utilisateur IAM.

Amazon ECS prend en charge les clés de condition spécifiques au service suivantes que vous pouvez utiliser pour fournir un filtrage précis pour vos politiques IAM :

Clé de condition	Description	Types d'évaluation
aws :RequestTag/$ {TagKey}	La clé de contexte est formatée `"aws:RequestTag/ tag-key":"tag-value "` où *tag-key* et *tag-value* sont une paire clé de balise et valeur. Vérifie que la paire clé-valeur de la balise est présente dans un AWS demande. Par exemple, vous pouvez vérifier que la requête comprend la clé d'étiquette `"Dept"` et qu'elle a la valeur `"Accounting"` .	Chaîne
aws :ResourceTag/$ {TagKey}	La clé de contexte est formatée `"aws:ResourceTag/ tag-key":"tag-value "` où *tag-key* et *tag-value* sont une paire clé de balise et valeur. Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	Chaîne
aws :TagKeys	Cette clé de contexte est formatée `"aws:TagKeys":" tag-key"` où *tag-key* est une liste de clés de tag sans valeurs (par exemple, `["Dept","Cost-Center"]`). Vérifie les clés de tag présentes dans un AWS demande.	Chaîne
ecos :ResourceTag/$ {TagKey}	La clé de contexte est formatée `"ecs:ResourceTag/ tag-key":"tag-value "` où *tag-key* et *tag-value* sont une paire clé de balise et valeur.	Chaîne

Clé de condition	Description	Types d'évaluation
	Vérifie que l'étiquette attachée à la ressource d'identité (utilisateur ou rôle) correspond aux nom et valeur de la clé spécifiée.	
ecs:account-setting	La clé de contexte est formatée "`ecs:account-setting":"` *`account-setting`* `"` où *`account-setting`* est le nom du paramètre du compte.	Chaîne
ecs:auto-assign-public-ip	La clé de contexte est formatée "`ecs:auto-assign-public-ip":"` *`value`*" où *`value-`* est »`true`« ou »`false`«.	Chaîne
ecs:capacity-provider	La clé de contexte est formatée "`ecs:capacity-provider":"` *`capacity-provider-arn`* `"` où *`capacity-provider-arn`* est l'ARN du fournisseur de capacité.	ARN, Null
ecs:cluster	La clé de contexte est formatée "`ecs:cluster":"` *`cluster-arn`* `"` où *`cluster-arn`* est l'ARN du cluster Amazon ECS.	ARN, Null
ecs:compute-compatability	La clé de contexte est formatée "`ecs:compute-compatability":"` *`compatability-type`* `"` où *`compatability-type`* est »`FARGATE`«, »`EC2`« ou »`EXTERNAL`«.	Chaîne
ecs:container-instances	La clé de contexte est formatée "`ecs:container-instances":"` *`container-instance-arns`* `"` où *`container-instance-arns`* est un ou plusieurs ARN d'instance de conteneur.	ARN, Null
ecs:container-name	La clé de contexte est formatée "`ecs:container-name":"` *`container-name`* `"` où *`container-instance-`* est le nom d'un conteneur Amazon ECS défini dans la définition de la tâche.	Chaîne

Clé de condition	Description	Types d'évaluation
ecs:enable-execute-command	La clé de contexte est formatée "ecs:enable-execute-command":" *value*" où *value-* est »true« ou »false«.	Chaîne
ecs:enable-service-connect	La clé de contexte est formatée "ecs:enable-service-connect":" *value*" où *value* est "true" ou "false".	Chaîne
ecs:enable-ebs-volumes	La clé de contexte est formatée "ecs:enable-ebs-volumes":" *value*" où *value* est "true" ou "false".	Chaîne
ecs:enable-managed-tags	La clé de contexte est formatée "ecs:enable-managed-tags":" *value*" où *value* est "true" ou "false".	Chaîne
ecs:enable-vpc-lattice	La clé de contexte est formatée "ecs:vpc-lattice":" *value*" où *value* est "true" ou "false".	Chaîne
ecs:fargate-ephemeral-storage-kms-key	La clé de contexte est formatée "ecs:fargate-ephemeral-storage-kms-key":" *key*" où *key* est l'identifiant du AWS KMS clé.	Chaîne
ecs:namespace	La clé de contexte est formatée "ecs:namespace":" *namespace-arn* " où *namespace-arn* est l'ARN du AWS Cloud Map espace de noms.	ARN, Null
ecs:propagate-tags	La clé de contexte est formatée "ecs:propagate-tags":" *value*" où *value* est "TASK_DEFINITION" , "SERVICE" , ou "NONE".	Chaîne
ecs:service	La clé de contexte est formatée "ecs:service":" *service-arn* " où *service-arn* est l'ARN du service Amazon ECS.	ARN, Null

Clé de condition	Description	Types d'évaluation
ecs:task-definition	La clé de contexte est formatée "ecs:task-definition":" *task-definition-arn* " où *task-definition-arn* est l'ARN de la définition de tâche Amazon ECS.	ARN, Null
ecs:subnet	La clé de contexte est formatée "ecs:subnet":" *subnets*" où *subnets* sont les sous-réseaux d'une tâche ou d'un service qui utilise awsvpc mode.	Chaîne
ecs:task	La clé de contexte est formatée "ecs:task":" *task-arn*" où *task-arn* est l'ARN du service Amazon ECS.	ARN, Null
ecs:task-cpu	La clé de contexte est formatée "ecs:task-cpu":"*task-cpu*" où *task-cpu* est le processeur de la tâche, sous la forme d'un entier avec 1024 = 1 vCPU.	Entier
ecs:task-memory	La clé de contexte est formatée "ecs:task-memory":" *task-memory* " où *task-memory* est la mémoire des tâches en MiB.	Entier

Pour voir la liste des clés de condition Amazon ECS, consultez Clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service. Pour savoir avec quelles actions et ressources vous pouvez utiliser une clé de condition, consultez Actions définies par Amazon Elastic Container Service.

Pour voir des exemples de politiques Amazon ECS basées sur l'identité, consultez Identity-based exemples de politiques pour Amazon Elastic Container Service.

Listes de contrôle d'accès (ACL) dans Amazon ECS

Prend en charge les ACL : non

Les listes de contrôle d’accès (ACL) vérifie quels principaux (membres de compte, utilisateurs ou rôles) ont l’autorisation d’accéder à une ressource. Les listes de contrôle d’accès sont similaires aux politiques basées sur les ressources, bien qu’elles n’utilisent pas le format de document de politique JSON.

Attribute-based contrôle d'accès (ABAC) avec Amazon ECS

> **Important**
>
> Amazon ECS prend en charge le contrôle d'accès basé sur les attributs pour toutes les ressources Amazon ECS. Pour déterminer si vous pouvez utiliser des attributs pour définir le périmètre d'une action, utilisez le tableau Actions définies par Amazon ECS dans la Référence de l'autorisation de service. Vérifiez d'abord qu'il existe une ressource dans la colonne Ressource. Ensuite, utilisez le Clés de condition colonne pour voir les touches du action/resource combinaison.

Prise en charge d’ABAC (balises dans les politiques) : Oui

Attribute-based le contrôle d'accès (ABAC) est une stratégie d'autorisation qui définit les autorisations en fonction d'attributs appelés balises. Vous pouvez associer des balises aux entités IAM et AWS resources, puis concevez des politiques ABAC pour autoriser les opérations lorsque la balise du principal correspond à la balise de la ressource.

Pour contrôler l’accès basé sur des étiquettes, vous devez fournir les informations d’étiquette 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 plus d’informations sur ABAC, consultez Définition d’autorisations avec l’autorisation ABAC dans le Guide de l’utilisateur IAM. Pour accéder à un didacticiel décrivant les étapes de configuration de l’ABAC, consultez Utilisation du contrôle d’accès par attributs (ABAC) dans le Guide de l’utilisateur IAM.

Pour en savoir plus sur l'identification des ressources Amazon ECS, consultez Balisage des ressources Amazon ECS.

Pour visualiser un exemple de politique basée sur l'identité permettant de limiter l'accès à une ressource en fonction des étiquettes de cette ressource, consultez Description des services Amazon ECS basée sur des balises.

Utilisation d'informations d'identification temporaires avec Amazon ECS

Prend en charge les informations d’identification temporaires : oui

Les informations d'identification temporaires fournissent un accès de courte durée à AWS ressources et sont automatiquement créés lorsque vous utilisez la fédération ou changez de rôle. AWS recommande de générer dynamiquement des informations d'identification temporaires au lieu d'utiliser des clés d'accès à long terme. Pour plus d’informations, consultez Informations d’identification de sécurité temporaires dans IAM et Services AWS compatibles avec IAM dans le Guide de l’utilisateur IAM.

Transmission des sessions d’accès pour Amazon ECS

Prend en charge les sessions d’accès direct (FAS) : oui

Les sessions d'accès direct (FAS) utilisent les autorisations du principal appelant un Service AWS, combiné à la demande Service AWS pour adresser des demandes aux services en aval. Pour plus de détails sur la politique relative à la transmission de demandes FAS, consultez la section Sessions de transmission d’accès.

Fonctions du service pour Amazon ECS

Prend en charge les rôles de service : oui

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.

> **Warning**
>
> La modification des autorisations d'une fonction de service peut altérer la fonctionnalité d'Amazon ECS. Ne modifiez des rôles de service que quand Amazon ECS vous le conseille.

Service-linked rôles pour Amazon ECS

Prend en charge les rôles liés à un service : oui

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 AWS et appartiennent au service. Un administrateur IAM peut consulter, mais ne peut pas modifier, les autorisations concernant les rôles liés à un service.

Pour plus d'informations sur la création ou la gestion des rôles liés à un service Amazon ECS service, veuillez consulter Utilisation des rôles liés à un service pour Amazon ECS.

Identity-based exemples de politiques pour Amazon Elastic Container Service

Par défaut, les utilisateurs et les rôles ne sont pas autorisés à créer ou à modifier les ressources Amazon ECS. Pour octroyer aux utilisateurs des autorisations d'effectuer des actions sur les ressources dont ils ont besoin, un administrateur IAM peut créer des politiques IAM.

Pour apprendre à créer une politique basée sur l'identité IAM à l'aide de ces exemples de documents de politique JSON, consultez Création de politiques IAM (console) dans le Guide de l'utilisateur IAM.

Pour plus de détails sur les actions et les types de ressources définis par Amazon ECS, y compris le format des ARN pour chacun des types de ressources, consultez Actions, ressources et clés de condition pour Amazon Elastic Container Service dans la Référence de l'autorisation de service.

Rubriques

- Pratiques exemplaires en matière de politiques Amazon ECS
- Autorisation aux utilisateurs IAM d'afficher leurs propres autorisations
- Exemples de cluster Amazon ECS
- Exemples d'instances de conteneurs Amazon ECS
- Exemples de définition de tâche Amazon ECS
- Exemple d'exécution de tâche Amazon ECS
- Exemple de démarrage de tâche Amazon ECS
- Exemples de recensement et description de tâche Amazon ECS
- Exemple de création d'un service Amazon ECS
- Description des services Amazon ECS basée sur des balises
- Exemple de refus de remplacement de l'espace de noms Service Connect Amazon ECS

Pratiques exemplaires en matière de politiques Amazon ECS

Identity-based les politiques déterminent si quelqu'un peut créer des ressources Amazon ECS sur votre compte, y accéder ou les supprimer. Ces actions peuvent entraîner des frais pour votre Compte AWS. Lorsque vous créez ou modifiez des politiques basées sur l'identité, suivez ces instructions et recommandations :

- Commencez avec AWS politiques gérées et évolution vers les autorisations de moindre privilège — Pour commencer à accorder des autorisations à vos utilisateurs et à vos charges de travail, utilisez AWS politiques gérées qui accordent des autorisations pour de nombreux cas d'utilisation courants. Ils sont disponibles dans votre Compte AWS. Nous vous recommandons de réduire davantage les autorisations en définissant AWS des politiques gérées par les clients qui sont spécifiques à vos cas d'utilisation. Pour plus d'informations, consultez politiques gérées par AWS ou politiques gérées par AWS pour les activités professionnelles dans le Guide de l'utilisateur IAM.
- Accordez les autorisations de moindre privilège : lorsque vous définissez des autorisations avec des politiques 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. Pour plus d'informations sur l'utilisation d'IAM pour appliquer des autorisations, consultez politiques et autorisations dans IAM dans le Guide de l'utilisateur IAM.
- Utilisez des conditions dans les politiques IAM pour restreindre davantage l'accès : vous pouvez ajouter une condition à vos politiques afin de limiter l'accès aux actions et aux ressources. Par exemple, vous pouvez écrire une condition de politique pour spécifier que toutes les demandes doivent être envoyées via SSL. Vous pouvez également utiliser des conditions pour accorder l'accès aux actions de service si elles sont utilisées via un Service AWS, tels que CloudFormation. Pour plus d'informations, consultez Conditions pour éléments de politique JSON IAM dans le Guide de l'utilisateur IAM.
- Utilisez l'Analyseur d'accès IAM pour valider vos politiques IAM afin de garantir des autorisations sécurisées et fonctionnelles : l'Analyseur d'accès IAM valide les politiques nouvelles et existantes de manière à ce que les politiques IAM respectent le langage de politique IAM (JSON) et les bonnes pratiques IAM. IAM Access Analyzer fournit plus de 100 vérifications de politiques et des recommandations exploitables pour vous aider à créer des politiques sécurisées et fonctionnelles. Pour plus d'informations, consultez Validation de politiques avec IAM Access Analyzer dans le Guide de l'utilisateur IAM.
- Exiger une authentification multifactorielle (MFA) — Si vous avez un scénario qui nécessite des utilisateurs IAM ou un utilisateur root dans votre Compte AWS, activez l'authentification multifacteur

pour renforcer la sécurité. Pour exiger la MFA lorsque des opérations d'API sont appelées, ajoutez des conditions MFA à vos politiques. Pour plus d'informations, consultez Sécurisation de l'accès aux API avec MFA dans le Guide de l'utilisateur IAM.

Pour plus d'informations sur les bonnes pratiques dans IAM, consultez Bonnes pratiques de sécurité dans IAM dans le Guide de l'utilisateur IAM.

Autorisation aux utilisateurs IAM d'afficher leurs propres autorisations

Cet exemple montre comment créer une politique qui permet aux utilisateurs IAM d'afficher les politiques en ligne et gérées attachées à leur identité d'utilisateur. Cette politique inclut les autorisations nécessaires pour effectuer cette action sur la console ou par programmation à l'aide du AWS CLI ou AWS API.

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Sid": "ViewOwnUserInfo",
            "Effect": "Allow",
            "Action": [
                "iam:GetUserPolicy",
                "iam:ListGroupsForUser",
                "iam:ListAttachedUserPolicies",
                "iam:ListUserPolicies",
                "iam:GetUser"
            ],
            "Resource": ["arn:aws:iam::*:user/${aws:username}"]
        },
        {
            "Sid": "NavigateInConsole",
            "Effect": "Allow",
            "Action": [
                "iam:GetGroupPolicy",
                "iam:GetPolicyVersion",
                "iam:GetPolicy",
                "iam:ListAttachedGroupPolicies",
                "iam:ListGroupPolicies",
                "iam:ListPolicyVersions",
                "iam:ListPolicies",
                "iam:ListUsers"
            ],
```

```
            "Resource": "*"
        }
    ]
}
```

Exemples de cluster Amazon ECS

La stratégie IAM suivante accorde l'autorisation de créer et de répertorier des clusters. Les actions `CreateCluster` et `ListClusters` n'acceptent aucune ressource. La définition de ressource est donc définie sur * pour toutes les ressources.

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateCluster",
                "ecs:ListClusters"
            ],
            "Resource": "*"
        }
    ]
}
```

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DescribeClusters",
                "ecs:DeleteCluster"
            ],
            "Resource": ["arn:aws:ecs:us-east-1:123456789012:cluster/cluster-
name"]
        }
    ]
```

```
}
```

La stratégie IAM suivante accorde l'autorisation de décrire et de supprimer un cluster spécifique. Les actions `DescribeClusters` et `DeleteCluster` acceptent les ARN de cluster en tant que ressources.

JSON

```
{
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
```

```
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:123456789012:cluster/default"}
            }
        }
    ]
}
```

La stratégie IAM suivante peut être liée à un utilisateur ou à groupe, ce qui autoriserait cet utilisateur ou ce groupe à effectuer des opérations uniquement sur un cluster spécifique.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:Describe*",
                "ecs:List*"
            ],
            "Effect": "Allow",
            "Resource": "*"
        },
        {
            "Action": [
                "ecs:DeleteCluster",
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances",
                "ecs:RegisterContainerInstance",
                "ecs:SubmitContainerStateChange",
                "ecs:SubmitTaskStateChange"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:111122223333:cluster/default"
        },
        {
            "Action": [
                "ecs:DescribeContainerInstances",
                "ecs:DescribeTasks",
```

```
                "ecs:ListTasks",
                "ecs:UpdateContainerAgent",
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Effect": "Allow",
            "Resource": "*",
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:111122223333:cluster/default"}
            }
        }
    ]
}
```

Exemples d’instances de conteneurs Amazon ECS

L'enregistrement des instances de conteneur est géré par l'agent Amazon ECS, mais vous pouvez parfois avoir besoin d'autoriser un utilisateur à annuler manuellement l'enregistrement d'une instance d'un cluster. Il est possible que l'instance de conteneur ait été enregistrée accidentellement dans le mauvais cluster ou que l'instance ait été résiliée alors que des tâches étaient en cours d'exécution.

La stratégie IAM suivante permet à un utilisateur de répertorier et d'annuler l'enregistrement des instances de conteneur qui se trouvent dans un cluster donné :

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:DeregisterContainerInstance",
                "ecs:ListContainerInstances"
            ],
            "Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/cluster_name"
        }
    ]
}
```

Exemples de définition de tâche Amazon ECS

Les politiques IAM de définition de tâche ne prennent pas en charge les autorisations au niveau des ressources, mais la politique IAM suivante autorise un utilisateur à enregistrer, répertorier et décrire des définitions de tâche :

Si vous utilisez la console, vous devez ajouter `CloudFormation: CreateStack` en tant qu'`Action`.

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:RegisterTaskDefinition",
                "ecs:ListTaskDefinitions",
                "ecs:DescribeTaskDefinition"
            ],
            "Resource": "*"
        }
    ]
}
```

Exemple d’exécution de tâche Amazon ECS

Les ressources pour `RunTask` sont des définitions de tâches. Pour limiter les clusters sur lesquels un utilisateur peut exécuter des définitions de tâches, vous pouvez les spécifier dans le bloc `Condition`. Cette méthode présente l'avantage de ne pas avoir besoin de répertorier les définitions de tâche et les clusters de vos ressources pour accorder un accès approprié. Vous pouvez appliquer l'une ou l'autre, ou les deux.

La politique IAM suivante accorde l'autorisation d'exécuter toute révision d'une définition de tâche spécifique sur un cluster spécifique :

JSON

```
{
```

```
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:123456789012:cluster/cluster_name"}
            },
            "Resource": ["arn:aws:ecs:us-east-1:123456789012:task-definition/
task_family:*"]
        }
    ]
}
```

Exemple de démarrage de tâche Amazon ECS

Les ressources pour `StartTask` sont des définitions de tâches. Pour limiter les clusters et les instances de conteneur sur lesquels un utilisateur peut démarrer des définitions de tâches, vous pouvez les spécifier dans le bloc `Condition`. Cette méthode présente l'avantage de ne pas avoir besoin de répertorier les définitions de tâche et les clusters de vos ressources pour accorder un accès approprié. Vous pouvez appliquer l'une ou l'autre, ou les deux.

La stratégie IAM suivante accorde l'autorisation de lancer la révision d'une définition de tâche spécifique sur un cluster spécifique et une instance de conteneur spécifique.

> **Note**
>
> Dans cet exemple, lorsque vous appelez le `StartTask` API avec AWS CLI ou autre AWS SDK, vous devez spécifier la révision de la définition de tâche afin que `Resource` correspondance cartographique.

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:StartTask"],
            "Condition": {
```

```
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:us-
east-1:123456789012:cluster/cluster_name",
                    "ecs:container-instances": ["arn:aws:ecs:us-
east-1:123456789012:container-instance/cluster_name/container_instance_UUID"]
                }
            },
            "Resource": ["arn:aws:ecs:us-east-1:123456789012:task-
definition/task_family:*"]
        }
    ]
}
```

Exemples de recensement et description de tâche Amazon ECS

La stratégie IAM suivante permet à un utilisateur de décrire une tâche spécifiée dans un cluster spécifié :

JSON

```
{
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:DescribeTasks"],
            "Condition": {
                "ArnEquals": {"ecs:cluster": "arn:aws:ecs:us-
east-1:123456789012:cluster/cluster_name"}
            },
            "Resource": ["arn:aws:ecs:us-
east-1:123456789012:task/cluster_name/task_UUID"]
        }
    ]
}
```

Exemple de création d’un service Amazon ECS

La stratégie IAM suivante permet à un utilisateur de créer des services Amazon ECS dans la Console de gestion AWS :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "application-autoscaling:Describe*",
                "application-autoscaling:PutScalingPolicy",
                "application-autoscaling:RegisterScalableTarget",
                "cloudwatch:DescribeAlarms",
                "cloudwatch:PutMetricAlarm",
                "ecs:List*",
                "ecs:Describe*",
                "ecs:CreateService",
                "elasticloadbalancing:Describe*",
                "iam:GetPolicy",
                "iam:GetPolicyVersion",
                "iam:GetRole",
                "iam:ListAttachedRolePolicies",
                "iam:ListRoles",
                "iam:ListGroups",
                "iam:ListUsers"
            ],
            "Resource": "*"
        }
    ]
}
```

Description des services Amazon ECS basée sur des balises

Vous pouvez utiliser des conditions dans votre stratégie basée sur l'identité pour contrôler l'accès aux ressources Amazon ECS en fonction des balises. Cet exemple montre comment créer une politique qui autorise la description de vos services. Toutefois, l'autorisation est accordée uniquement si l'étiquette de service `Owner` a la valeur du nom d'utilisateur de cet utilisateur. Cette politique accorde également les autorisations nécessaires pour réaliser cette action sur la console.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "DescribeServices",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "*"
        },
        {
            "Sid": "ViewServiceIfOwner",
            "Effect": "Allow",
            "Action": "ecs:DescribeServices",
            "Resource": "arn:aws:ecs:*:*:service/*",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Owner": "${aws:username}"}
            }
        }
    ]
}
```

Vous pouvez rattacher cette politique aux utilisateurs IAM de votre compte. Si un utilisateur nommé `richard-roe` tente de décrire un service Amazon ECS service, le service doit être balisé `Owner=richard-roe` ou `owner=richard-roe`. Dans le cas contraire, l'utilisateur se voit refuser l'accès. La clé de condition d'étiquette `Owner` correspond à la fois à `Owner` et à `owner`, car les noms de clé de condition ne sont pas sensibles à la casse. Pour plus d'informations, veuillez consulter la rubrique Éléments de stratégie JSON IAM : Condition dans le Guide de l'utilisateur IAM.

Exemple de refus de remplacement de l’espace de noms Service Connect Amazon ECS

La politique IAM suivante interdit à un utilisateur de remplacer l'espace de noms Service Connect par défaut dans une configuration de service. L'espace de noms par défaut est défini dans le cluster. Cependant, vous pouvez le remplacer dans une configuration de service. Par souci de cohérence, pensez à configurer tous vos nouveaux services pour qu'ils utilisent le même espace de noms. Utilisez les clés de contexte suivantes pour demander aux services d'utiliser un espace

de noms spécifique. Remplacez la <region>, le <aws_account_id>, le <cluster_name> et l'<namespace_id> par les vôtres dans l'exemple suivant :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:CreateService",
                "ecs:UpdateService"
            ],
            "Condition": {
                "ArnEquals": {
                    "ecs:cluster": "arn:aws:ecs:us-
east-1:123456789012:cluster/cluster_name",
                    "ecs:namespace": "arn:aws:servicediscovery:us-
east-1:123456789012:namespace/namespace_id"
                }
            },
            "Resource": "*"
        }
    ]
}
```

AWS politiques gérées pour Amazon Elastic Container Service

Pour ajouter des autorisations aux utilisateurs, aux groupes et aux rôles, il est plus facile d'utiliser AWS politiques gérées plutôt que de rédiger vous-même des politiques. 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 démarrer rapidement, vous pouvez utiliser notre AWS politiques gérées. Ces politiques couvrent les cas d'utilisation courants et sont disponibles dans votre AWS compte. Pour plus d'informations sur AWS politiques gérées, voir AWS politiques gérées dans le Guide de l'utilisateur IAM.

AWS maintenance et mise à jour des services AWS politiques gérées. Vous ne pouvez pas modifier les autorisations dans AWS politiques gérées. Les services ajoutent parfois des autorisations supplémentaires à un AWS politique gérée pour prendre en charge les nouvelles fonctionnalités. Ce type de mise à jour affecte toutes les identités (utilisateurs, groupes et rôles) auxquelles la politique est attachée. Les services sont les plus susceptibles de mettre à jour un AWS politique gérée lorsqu'une nouvelle fonctionnalité est lancée ou lorsque de nouvelles opérations deviennent disponibles. Les services ne suppriment pas les autorisations d'un AWS politique gérée, afin que les mises à jour des politiques n'enfreignent pas vos autorisations existantes.

En outre, AWS prend en charge les politiques gérées pour les fonctions professionnelles qui couvrent plusieurs services. Par exemple, le ReadOnlyAccess AWS la politique gérée fournit un accès en lecture seule à tous AWS services et ressources. Lorsqu'un service lance une nouvelle fonctionnalité, AWS ajoute des autorisations en lecture seule pour les nouvelles opérations et ressources. Pour obtenir la liste des politiques de fonctions professionnelles et leurs descriptions, consultez la page politiques gérées par AWS pour les fonctions de tâche dans le Guide de l'utilisateur IAM.

Amazon ECS et Amazon ECR fournissent plusieurs stratégies gérées et relations d'approbation que vous pouvez attacher à des utilisateurs, groupes, rôles, des instances Amazon EC2 et des tâches Amazon ECS qui autorisent différents niveaux de contrôle sur les ressources et les opérations d'API. Vous pouvez appliquer ces politiques directement ou les utiliser comme points de départ pour créer vos propres politiques. Pour plus d'informations sur les stratégies gérées par Amazon ECR, veuillez consulter Stratégies gérées par Amazon ECR.

Amazon ECS_FullAccess

Vous pouvez associer la politique `AmazonECS_FullAccess` à vos identités IAM. Cette politique accorde un accès administratif aux ressources Amazon ECS et accorde à une identité IAM (telle qu'un utilisateur, un groupe ou un rôle) l'accès au AWS services auxquels Amazon ECS est intégré pour utiliser toutes les fonctionnalités d'Amazon ECS. L'utilisation de cette stratégie permet d'accéder à toutes les fonctions Amazon ECS disponibles dans la Console de gestion AWS.

Pour consulter les autorisations relatives à cette politique, consultez Amazon ECS_FullAccess dans le AWS Référence des politiques gérées.

AmazonECSInfrastructureRolePolicyForVolumes

Vous pouvez associer la politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` à vos entités IAM.

La politique accorde les autorisations nécessaires à Amazon ECS pour AWS Appels d'API en votre nom. Vous pouvez associer cette politique au rôle IAM que vous fournissez avec la configuration de votre volume lorsque vous lancez des tâches et des services Amazon ECS. Ce rôle permet à Amazon ECS de gérer les volumes associés à vos tâches. Pour de plus amples informations, consultez la section Rôle IAM d'infrastructure Amazon ECS.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRolePolicyForVolumes dans le AWS Guide de référence des stratégies gérées par.

AmazonEC2ContainerServiceforEC2Role

Vous pouvez associer la politique `AmazonEC2ContainerServiceforEC2Role` à vos identités IAM. Cette politique accorde des autorisations administratives qui permettent aux instances de conteneurs Amazon ECS de passer des appels à AWS en votre nom. Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.

Amazon ECS associe à cette politique une fonction de service qui permet à Amazon ECS d'effectuer des actions en votre nom sur des instances Amazon EC2 ou des instances externes.

Pour voir les autorisations de cette stratégie, consultez AmazonEC2ContainerServiceforEC2Role dans le AWS Guide de référence des stratégies gérées par.

Considérations

Notez également les points suivants lorsque vous utilisez une politique IAM gérée par `AmazonEC2ContainerServiceforEC2Role`.

- En suivant les conseils de sécurité standard pour accorder le moindre privilège, vous pouvez modifier la stratégie IAM gérée par `AmazonEC2ContainerServiceforEC2Role` pour répondre à vos besoins spécifiques. Si l'une des autorisations accordées dans la stratégie gérée n'est pas nécessaire pour votre cas d'utilisation, créez une stratégie personnalisée et ajoutez uniquement les autorisations dont vous avez besoin. Par exemple, l'autorisation `UpdateContainerInstancesState` est fournie pour le drainage d'instances Spot. Si cette autorisation n'est pas nécessaire pour votre cas d'utilisation, excluez-la à l'aide d'une stratégie personnalisée.
- Les conteneurs qui s'exécutent sur vos instances de conteneur ont accès à toutes les autorisations fournies au rôle d'instance de conteneur par le biais des métadonnées d'instance. Nous vous recommandons de limiter les autorisations dans votre rôle d'instance de conteneur à la liste minimale d'autorisations fournie dans la stratégie gérée par

`AmazonEC2ContainerServiceforEC2Role` présentée ci-dessous. Si les conteneurs dans vos tâches ont besoin d'autorisations supplémentaires qui ne sont pas répertoriées ici, nous vous recommandons de fournir leurs propres rôles IAM à ces tâches. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

Vous pouvez empêcher les conteneurs du bridge `docker0` d'accéder aux autorisations fournies au rôle de l'instance de conteneur, et ce tout en autorisant les autorisations fournies par rôle IAM de tâche Amazon ECS en exécutant la commande iptables suivante sur vos instances de conteneur. Les conteneurs ne peuvent pas interroger les métadonnées d'instance lorsque cette règle est en vigueur. Notez que cette commande suppose une configuration bridge Docker par défaut et ne fonctionnera pas avec les conteneurs qui utilisent le mode réseau `host`. Pour plus d'informations, consultez Mode réseau.

```
sudo yum install -y iptables-services; sudo iptables --insert DOCKER USER 1 --in-interface docker+ --destination 169.254.169.254/32 --jump DROP
```

Pour garantir le maintien de la règle iptables après un redémarrage, vous devez l'enregistrer sur votre instance de conteneur. Pour l'Amazon ECS-optimized AMI, utilisez la commande suivante. Pour les autres systèmes d'exploitation, consultez la documentation correspondante.

- Pour l'Amazon ECS-optimized AMI Amazon Linux 2 :

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now iptables
  ```

- Pour l'Amazon ECS-optimized AMI Amazon Linux :

  ```
  sudo service iptables save
  ```

AmazonEC2ContainerServiceEventsRole

Vous pouvez associer la politique `AmazonEC2ContainerServiceEventsRole` à vos identités IAM. Cette politique accorde des autorisations qui permettent à Amazon EventBridge (anciennement CloudWatch Événements) pour exécuter des tâches en votre nom. Cette stratégie peut être attachée au rôle IAM spécifié lorsque vous créez des tâches planifiées. Pour plus d'informations, consultez Amazon ECS EventBridge Rôle IAM.

Pour voir les autorisations de cette stratégie, consultez AmazonEC2ContainerServiceEventsRole dans le AWS Guide de référence des stratégies gérées par.

AmazonECSTaskExecutionRolePolicy

Le `AmazonECSTaskExecutionRolePolicy` la politique IAM gérée accorde les autorisations nécessaires à l'agent de conteneur Amazon ECS et AWS Fargate agents de conteneurs à fabriquer AWS Appels d'API en votre nom. Cette stratégie peut être ajoutée à votre rôle IAM d'exécution de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Pour voir les autorisations de cette stratégie, consultez AmazonECSTaskExecutionRolePolicy dans le AWS Guide de référence des stratégies gérées par.

AmazonECSServiceRolePolicy

La politique IAM gérée `AmazonECSServiceRolePolicy` permet à Amazon Elastic Container Service de gérer votre cluster. Cette politique peut être ajoutée à votre AWSServiceRoleForECS rôle lié à un service.

Pour voir les autorisations de cette stratégie, consultez AmazonECSServiceRolePolicy dans le AWS Guide de référence des stratégies gérées par.

`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecu`

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` à vos entités IAM. Cette politique accorde un accès administratif à Autorité de certification privée AWS, Secrets Manager et autres AWS Services requis pour gérer les fonctionnalités TLS d'Amazon ECS Service Connect en votre nom.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity dans le AWS Guide de référence des stratégies gérées par.

`AWSApplicationAutoscalingECSServicePolicy`

Vous ne pouvez pas joindre de `AWSApplicationAutoscalingECSServicePolicy` à vos entités IAM. Cette stratégie est attachée à un rôle lié à un service qui permet à Application Auto Scaling d'effectuer des actions en votre nom. Pour plus d'informations, voir Service-linked rôles pour Application Auto Scaling.

Pour voir les autorisations de cette stratégie, consultez AWSApplicationAutoscalingECSServicePolicy dans le AWS Guide de référence des stratégies gérées par.

AWSCodeDeployRoleForECS

Vous ne pouvez pas joindre de `AWSCodeDeployRoleForECS` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui permet CodeDeploy pour effectuer des actions en votre nom. Pour plus d'informations, voir Créez un rôle de service pour CodeDeploy dans le AWS CodeDeploy Guide de l'utilisateur.

Pour voir les autorisations de cette stratégie, consultez AWSCodeDeployRoleForECS dans le AWS Guide de référence des stratégies gérées par.

AWSCodeDeployRoleForECSLimited

Vous ne pouvez pas joindre de `AWSCodeDeployRoleForECSLimited` à vos entités IAM. Cette politique est associée à un rôle lié à un service qui permet CodeDeploy pour effectuer des actions en votre nom. Pour plus d'informations, voir Créez un rôle de service pour CodeDeploy dans le AWS CodeDeploy Guide de l'utilisateur.

Pour voir les autorisations de cette stratégie, consultez AWSCodeDeployRoleForECSLimited dans le AWS Guide de référence des stratégies gérées par.

AmazonECSInfrastructureRolePolicyForLoadBalancers

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForLoadBalancers` à vos entités IAM. Cette politique accorde des autorisations qui permettent à Amazon ECS de gérer les ressources Elastic Load Balancing en votre nom. La politique inclut :

- Read-only autorisations pour décrire les auditeurs, les règles, les groupes cibles et l'état de santé de la cible
- les autorisations pour enregistrer et annuler l’enregistrement des cibles auprès de groupes cibles ;
- les autorisations de modification des écouteurs pour les Application Load Balancer et Network Load Balancer ;
- les autorisations permettant de modifier les règles pour les Application Load Balancer.

Ces autorisations permettent à Amazon ECS de gérer automatiquement les configurations de l’équilibreur de charge lors de la création ou de la mise à jour des services, garantissant ainsi le bon acheminement du trafic vers vos conteneurs.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRolePolicyForLoadBalancers dans le AWS Guide de référence des stratégies gérées par.

`AmazonECSInfrastructureRolePolicyForManagedInstances`

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForManagedInstances` à vos entités IAM. Cette politique accorde les autorisations requises par Amazon ECS pour créer et mettre à jour les ressources Amazon EC2 pour les instances gérées par ECS en votre nom. La politique inclut :

- les autorisations de création et de gestion de modèles de lancement Amazon EC2 pour les instances gérées ;
- Autorisations pour provisionner des instances Amazon EC2 à l'aide de CreateFleet et RunInstances
- les autorisations pour créer et gérer des balises sur les ressources Amazon EC2 créées par ECS ;
- les autorisations permettant de transmettre des rôles IAM aux instances Amazon EC2 pour les instances gérées ;
- les autorisations pour créer des rôles liés à un service pour les instances Spot Amazon EC2 ;
- Read-only autorisations pour décrire les ressources Amazon EC2, y compris les instances, les types d'instances, les modèles de lancement, les interfaces réseau, les zones de disponibilité, les groupes de sécurité, les sous-réseaux, les VPC, les images EC2 et les réservations de capacité
- Read-only autorisations pour mettre Amazon en vente ResourceGroups resources, qui nécessite des autorisations sous-jacentes pour obtenir des ressources balisées et répertorier Amazon CloudFormation empiler les ressources

Ces autorisations permettent à Amazon ECS d’allouer et de gérer automatiquement des instances Amazon EC2 pour vos instances gérées ECS, garantissant ainsi une configuration et une gestion du cycle de vie appropriées des ressources de calcul sous-jacentes.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRolePolicyForManagedInstances dans le AWS Guide de référence des stratégies gérées par.

AmazonECSInfrastructureRolePolicyForVpcLattice

Vous pouvez associer la politique `AmazonECSInfrastructureRolePolicyForVpcLattice` à vos entités IAM. Cette politique donne accès à d'autres AWS les ressources de service nécessaires pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRolePolicyForVpcLattice dans le AWS Guide de référence des stratégies gérées par.

Permet d'accéder à d'autres AWS ressources de service requises pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.

AmazonECSInfrastructureRoleforExpressGatewayServices

Vous pouvez associer la politique `AmazonECSInfrastructureRoleforExpressGatewayServices` à vos entités IAM. Cette politique accorde les autorisations requises par Amazon ECS pour créer et mettre à jour des applications Web à l'aide d'Express Services en votre nom. La politique inclut :

- Autorisations pour créer des rôles liés à des services pour Amazon ECS Application Auto Scaling
- Autorisations pour créer, modifier et supprimer des équilibreurs de charge d'applications, des écouteurs, des règles et des groupes cibles
- Autorisations pour créer, modifier et supprimer des groupes de sécurité VPC pour ECS-managed ressources
- Autorisations pour demander, gérer et supprimer SSL/TLS certificats via ACM
- Autorisations pour configurer les politiques et les cibles Application Auto Scaling pour les services Amazon ECS
- Autorisations de création et de gestion CloudWatch alarmes pour les déclencheurs de dimensionnement automatique
- Read-only autorisations pour décrire les équilibreurs de charge, les ressources VPC, les certificats, les configurations de dimensionnement automatique et CloudWatch alarmes

Ces autorisations permettent à Amazon ECS de provisionner et de gérer automatiquement les composants d'infrastructure requis pour les applications Web Express Services, notamment l'équilibrage de charge, les groupes de sécurité, les certificats SSL et les configurations de dimensionnement automatique.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInfrastructureRoleforExpressGatewayServices dans le AWS Guide de référence des stratégies gérées par.

`AmazonECSComputeServiceRolePolicy`

Le `AmazonECSComputeServiceRolePolicy` la politique est rattachée à la AmazonECSComputeServiceRole rôle lié à un service. Pour plus d'informations, consultez Utilisation de rôles pour gérer les instances gérées Amazon ECS.

Cette politique inclut les autorisations suivantes qui permettent à Amazon EKS d'effectuer les tâches suivantes :

- Amazon ECS peut décrire et supprimer des modèles de lancement.
- Amazon ECS peut décrire et supprimer les versions des modèles de lancement.
- Amazon ECS peut mettre fin à des instances.
- Amazon ECS peut décrire les paramètres de données d'instance suivants :
  - Instance
  - Interfaces réseau d'instance : Amazon ECS peut décrire les interfaces permettant de gérer le cycle de vie des instances EC2.
  - Fenêtre d'événements d'instance : Amazon ECS peut décrire les informations de la fenêtre d'événements afin de déterminer si le flux de travail peut être interrompu pour appliquer des correctifs à l'instance.
  - État de l'instance : Amazon ECS peut décrire l'état de l'instance afin de surveiller son état.

Pour voir les autorisations de cette stratégie, consultez AmazonECSComputeServiceRolePolicy dans le AWS Guide de référence des stratégies gérées par.

`AmazonECSInstanceRolePolicyForManagedInstances`

La politique `AmazonECSInstanceRolePolicyForManagedInstances` autorise les instances gérées Amazon ECS à s'enregistrer auprès des clusters Amazon ECS et à communiquer avec le service Amazon ECS.

Cette politique inclut les autorisations suivantes permettant à Amazon EKS d'effectuer les tâches suivantes :

- Enregistrement et annulation de l'enregistrement auprès des clusters Amazon ECS.

- Soumission des modifications d'état des instances de conteneur.
- Soumission des modifications de l'état des tâches.
- Découverte des points de terminaison de sondage pour l'agent Amazon ECS.

Pour voir les autorisations de cette stratégie, consultez AmazonECSInstanceRolePolicyForManagedInstances dans le AWS Guide de référence des stratégies gérées par.

Mises à jour d'Amazon ECS pour AWS stratégies gérées

Afficher les détails des mises à jour de AWS politiques gérées pour Amazon ECS depuis que ce service a commencé à suivre ces modifications. Pour recevoir des alertes automatiques sur les modifications apportées à cette page, abonnez-vous au flux RSS dans la page de l'historique des documents Amazon ECS.

Modifier	Description	Date
Ajoutez des autorisations à Amazon ECS_FullAccess.	Le `AmazonECS_FullAccess` la politique IAM gérée a été mise à jour pour inclure le `cloudformation:TagResource` et `cloudformation:UntagResource` autorisations. Ces autorisations permettent d'ajouter des balises à CloudFormation piles lorsqu'une pile est créée ou mise à jour.	2 septembre 2026
Ajout d'autorisations à AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour pour inclure l'autorisation `autoscaling:TerminateInstanceInAutoScalingGroup` . Cette	24 avril 2026

Modifier	Description	Date
	autorisation permet à Amazon ECS de mettre fin aux instances Amazon EC2 dans les ASG clients qui ne parviennent pas à s'enregis trer auprès du cluster Amazon ECS.	

Modifier	Description	Date
Mettre à jour `AmazonECSInfrastructureRolePolicyForManagedInstances` politique	Le `AmazonECSInfrastructureRolePolicyForManagedInstances` la politique a été mise à jour avec les autorisations suivantes pour prendre en charge les réservations de capacité sur les instances gérées Amazon ECS : • `resource-groups:ListGroupResources` l'autorisation a été ajoutée pour permettre à Amazon ECS de récupérer un groupe de réservations de capacité. Pour exécuter cette commande, les autorisations suivantes sont également requises : `cloudformation:DescribeStacks` , `cloudformation:ListStackResources` , `tag:GetResources` • `ec2:DescribeCapacityReservations` l'autorisation a été ajoutée pour permettre à Amazon ECS de recueillir des informations sur les réservations de capacité nécessaires à l'intégra	24 février 2026

Modifier	Description	Date
	tion aux instances gérées Amazon ECS. • `ec2:RunInstances` l'autorisation a été modifiée pour inclure Amazon ResourceGroups ressources de groupe permettant à Amazon ECS de fournir des réservations de capacité Amazon EC2 ciblées.	
Ajout d'autorisations à AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour pour inclure l'autorisation `ssmmessages:OpenDataChannel`. Cette autorisation permet à Amazon ECS d'ouvrir des canaux de données pour les sessions ECS Exec.	20 janvier 2026

Modifier	Description	Date
Mettre à jour `AmazonECSInfrastructureRolePolicyForManagedInstances` politique	Le `AmazonECSInfrastructureRolePolicyForManagedInstances` la politique a été mise à jour pour modifier la `CreateFleet` autorisations. Les conditions basées sur les ressources pour les sous-réseaux, les groupes de sécurité et les images EC2 ont été supprimées pour les raisons suivantes : • Les sous-réseaux et les groupes de sécurité ne sont ni créés ni gérés par Amazon ECS, ils ne contiennent donc pas les informations requises `AmazonECSManaged` balises. • Bien qu'Amazon ECS possède et gère les images EC2, les balises ne sont pas visibles pour les clients, ce qui empêche l'application de politiques basées sur les ressources. Cette modification garantit que la politique fonctionne correctement avec les ressources qui ne possèdent	15 décembre 2025

Modifier	Description	Date
	pas les balises de gestion ECS attendues.	
Ajout d'autorisations à AmazonECSServiceRolePolicy.	Le `AmazonECSServiceRolePolicy` la politique IAM gérée a été mise à jour avec de nouvelles autorisations Amazon EC2 qui permettent à Amazon ECS de récupérer les fenêtres d'événements Amazon EC2 pour les services et les clusters associés à Event Windows.	20 novembre 2025
Ajout d'autorisations à AmazonECSServiceRolePolicy.	Le `AmazonECSServiceRolePolicy` la politique IAM gérée a été mise à jour avec de nouvelles autorisations Amazon EC2 qui permettent à Amazon ECS de provisionner et de désactiver la tâche ENI.	14 novembre 2025

Modifier	Description	Date
Mettre à jour AmazonECS InfrastructureRolePolicyFor ServiceConnectTransportLayerSecurity politique	A mis à jour le AmazonECS InfrastructureRolePolicyFor ServiceConnectTransportLayerSecurity politique IAM gérée pour séparer le secretsmanager :DescribeSecret autorisation dans sa propre déclaration de politique. L'autorisation continue de limiter l'accès d'Amazon ECS exclusivement aux secrets créés par Amazon ECS et utilise la correspondance de modèles ARN au lieu de balises de ressources pour la définition de la portée. Cela permet à Amazon ECS de surveiller l'état des secrets tout au long de leur cycle de vie, y compris lorsqu'un secret a été supprimé.	13 novembre 2025
Ajout de la nouvelle politique the section called “AmazonECSInfrastructureRoleforExpressGatewayServices ”	Nouveau ajouté AmazonECS InfrastructureRoleforExpressGatewayServices politique qui permet à Amazon ECS d'accéder à la création et à la gestion d'applications Web à l'aide d'Express Services.	21 novembre 2025

Modifier	Description	Date
Ajout de la nouvelle politique the section called “AmazonECSInstanceRolePolicyForManagedInstances”	Nouveau ajouté AmazonECSInstanceRolePolicyForManagedInstances politique qui autorise les instances gérées par Amazon ECS à s'enregistrer auprès des clusters Amazon ECS.	30 septembre 2025
Ajout de la nouvelle politique the section called “AmazonECSInfrastructureRolePolicyForManagedInstances”	Nouveau ajouté AmazonECSInfrastructureRolePolicyForManagedInstances politique qui permet à Amazon ECS d'accéder à la création et à la gestion des ressources gérées par Amazon EC2.	30 septembre 2025
Ajout de la nouvelle politique AmazonECSComputeServiceRolePolicy	Permet à Amazon ECS de gérer vos instances gérées Amazon ECS et les ressources associées.	31 août 2025

Modifier	Description	Date
Ajouter des autorisations à the section called “AmazonEC2ContainerServiceforEC2Role” -	La politique IAM gérée `AmazonEC2ContainerServiceforEC2Role` a été mise à jour pour inclure l'autorisation `ecs:ListTagsForResource`. Cette autorisation permet à l'agent Amazon ECS de récupérer les balises des tâches et des instances de conteneurs via le point de terminaison des métadonnées de tâche (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`).	4 août 2025
Ajout d'autorisations à AmazonECSInfrastructureRolePolicyForLoadBalancers.	La politique IAM gérée `AmazonECSInfrastructureRolePolicyForLoadBalancers` a été mise à jour avec de nouvelles autorisations pour la description, l'annulation d'enregistrement et l'enregistrement des groupes cibles.	25 juillet 2025
Ajout de la nouvelle politique the section called “`AmazonECSInfrastructureRolePolicyForLoadBalancers`”	Nouveau ajouté AmazonECSInfrastructureRolePolicyForLoadBalancers politique qui donne accès à d'autres AWS ressources de service requises pour gérer les équilibreurs de charge associés aux charges de travail Amazon ECS.	15 juillet 2025

Modifier	Description	Date
Ajout d'autorisations à AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour avec de nouvelles autorisations AWS Cloud Map permettant à Amazon ECS de mettre à jour les attributs du service AWS Cloud Map pour les services gérés par Amazon ECS.	15 juillet 2025
Ajouter des autorisations à AmazonECSServiceRolePolicy	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour avec de nouvelles autorisations AWS Cloud Map permettant à Amazon ECS de mettre à jour les attributs du service AWS Cloud Map pour les services gérés par Amazon ECS.	24 juin 2025
Ajouter des autorisations à AmazonECSInfrastructureRolePolicyForVolumes	La politique `AmazonECSInfrastructureRolePolicyForVolumes` a été mise à jour pour ajouter l'autorisation `ec2:DescribeInstances`. Cette autorisation permet d'éviter les conflits de noms d'appareils pour les volumes Amazon EBS attachés à des tâches Amazon ECS exécutées sur la même instance de conteneur.	2 juin 2025

Modifier	Description	Date
Ajout de la nouvelle politique AmazonECSInfrastructureRolePolicyForVpcLattice	Permet d'accéder à d'autres AWS ressources de service requises pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.	18 novembre 2024
Ajouter des autorisations à AmazonECSInfrastructureRolePolicyForVolumes	La politique `AmazonECSInfrastructureRolePolicyForVolumes` a été mise à jour pour permettre aux clients de créer un volume Amazon EBS à partir d'un instantané.	10 octobre 2024
Autorisations ajoutées à the section called "Amazon ECS_FullAccess"	La politique `AmazonECS_FullAccess` a été mise à jour pour ajouter des autorisations `iam:PassRole` pour les rôles IAM pour un rôle nommé `ecsInfrastructureRole`. Il s'agit du rôle IAM par défaut créé par Console de gestion AWS qui est destiné à être utilisé comme rôle d'infrastructure ECS permettant à Amazon ECS de gérer les volumes Amazon EBS associés aux tâches ECS.	13 août 2024

Modifier	Description	Date
Ajout de la nouvelle politique AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity	Nouveau ajouté AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity politique qui fournit un accès administratif à AWS KMS, Autorité de certification privée AWS, Secrets Manager et permet aux fonctionnalités TLS d'Amazon ECS Service Connect de fonctionner correctement.	22 janvier 2024
Ajouter une nouvelle politique AmazonECSInfrastructureRolePolicyForVolumes	La politique `AmazonECSInfrastructureRolePolicyForVolumes` a été ajoutée. La politique accorde les autorisations nécessaires à Amazon ECS pour AWS Appels d'API pour gérer les volumes Amazon EBS associés aux charges de travail Amazon ECS.	11 janvier 2024
Ajouter des autorisations à AmazonECSServiceRolePolicy	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour avec de nouvelles autorisations `events` et des autorisations supplémentaires `autoscaling` et `autoscaling-plans` .	4 décembre 2023

Modifier	Description	Date
Ajouter des autorisations à AmazonEC2ContainerServiceEventsRole	Le `AmazonECSServiceRolePolicy` la politique IAM gérée a été mise à jour pour permettre l'accès au AWS Cloud Map `DiscoverInstancesRevision` Fonctionnement de l'API.	4 octobre 2023
Ajouter des autorisations à AmazonEC2ContainerServiceforEC2Role	La politique `AmazonEC2ContainerServiceforEC2Role` a été modifiée pour ajouter l'autorisation `ecs:TagResource` qui inclut une condition qui limite l'autorisation uniquement aux clusters nouvellement créés et aux instances de conteneur enregistrées.	06 mars 2023
Ajouter des autorisations à the section called “Amazon ECS_FullAccess”	La politique `AmazonECS_FullAccess` a été modifiée pour ajouter l'autorisation `elasticloadbalancing:AddTags` , qui inclut une condition qui limite l'autorisation uniquement aux équilibreurs de charge, aux groupes cibles, aux règles et aux écouteurs créés récemment. Cette autorisation n'autorise pas l'ajout de balises à des ressources Elastic Load Balancing déjà créées.	4 janvier 2023

Modifier	Description	Date
Amazon ECS a commencé à assurer le suivi des modifications	Amazon ECS a commencé à suivre les modifications apportées à ses AWS politiques gérées.	8 juin 2021

Absorption progressive AWS politiques IAM gérées pour Amazon Elastic Container Service

Les suivants AWS les politiques IAM gérées sont progressivement supprimées. Ces stratégies sont désormais remplacées par les stratégies mises à jour. Nous vous recommandons de mettre à jour vos utilisateurs ou rôles afin d'utiliser les politiques mises à jour.

AmazonEC2ContainerServiceFullAccess

> **Important**
>
> La politique IAM gérée `AmazonEC2ContainerServiceFullAccess` sera progressivement abandonnée à compter du 29 janvier 2021, en réponse à la constatation d'un problème de sécurité lié à l'autorisation `iam:passRole`. Cette autorisation autorise l'accès à toutes les ressources, y compris les informations d'identification aux rôles du compte. La politique étant progressivement abandonnée, vous ne pouvez pas attacher la politique à de nouveaux utilisateurs ou rôles. Tous les utilisateurs ou rôles auxquels la stratégie est déjà attachée peuvent continuer à l'utiliser. Toutefois, nous vous recommandons de mettre à jour vos utilisateurs ou rôles afin d'utiliser la politique gérée par `AmazonECS_FullAccess` à la place. Pour plus d'informations, consultez Migration vers le `Amazon ECS_FullAccess` politique gérée.

AmazonEC2ContainerServiceRole

> **Important**
>
> La politique IAM gérée `AmazonEC2ContainerServiceRole` sera progressivement abandonnée. Elle est désormais remplacée par le rôle lié au service Amazon ECS service. Pour plus d'informations, consultez Utilisation des rôles liés à un service pour Amazon ECS.

AmazonEC2ContainerServiceAutoscaleRole

> **Important**
>
> La stratégie IAM gérée `AmazonEC2ContainerServiceAutoscaleRole` sera progressivement abandonnée. Elle est désormais remplacée par le rôle lié au service Application Auto Scaling pour Amazon ECS service. Pour plus d'informations, voir Service-linked rôles pour Application Auto Scaling dans le Guide de l'utilisateur de l'application Auto Scaling.

Migration vers le **`Amazon ECS_FullAccess`** politique gérée

La stratégie IAM gérée `AmazonEC2ContainerServiceFullAccess` a été progressivement abandonnée à compter du 29 janvier 2021, en réponse à la constatation d'un problème de sécurité lié à l'autorisation `iam:passRole`. Cette autorisation autorise l'accès à toutes les ressources, y compris les informations d'identification aux rôles du compte. La politique étant progressivement abandonnée, vous ne pouvez pas attacher la politique à de nouveaux groupes, utilisateurs ou rôles. Tous les groupes, utilisateurs ou rôles auxquels la stratégie est déjà attachée peuvent continuer à l'utiliser. Toutefois, nous vous recommandons de mettre à jour vos groupes, utilisateurs ou rôles afin d'utiliser la politique gérée par `AmazonECS_FullAccess` à la place.

Les autorisations accordées par la politique `AmazonECS_FullAccess` incluent la liste complète des autorisations nécessaires pour utiliser ECS en tant qu'administrateur. Si vous utilisez actuellement des autorisations octroyées par la politique `AmazonEC2ContainerServiceFullAccess` qui ne sont pas dans la politique `AmazonECS_FullAccess`, vous pouvez les ajouter à une instruction de politique en ligne. Pour plus d'informations, consultez AWS politiques gérées pour Amazon Elastic Container Service.

Procédez comme suit pour déterminer si vous disposez de groupes, d'utilisateurs ou de rôles qui utilisent actuellement la politique IAM gérée par `AmazonEC2ContainerServiceFullAccess`. Ensuite, mettez-les à jour pour détacher la stratégie précédente et attachez la stratégie `AmazonECS_FullAccess`.

Pour mettre à jour un groupe, un utilisateur ou un rôle afin d'utiliser Amazon ECS_FullAccess politique (Console de gestion AWS)

1. Ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.

2. Dans le volet de navigation, choisissez Policies (Stratégies) et recherchez et sélectionnez la stratégie `AmazonEC2ContainerServiceFullAccess`.
3. Cliquez sur l'onglet Policy usage (Utilisation de la stratégie) qui affiche tout rôle IAM qui utilise actuellement cette stratégie.
4. Pour chaque rôle IAM qui utilise actuellement la politique `AmazonEC2ContainerServiceFullAccess`, sélectionnez le rôle et procédez comme suit pour détacher la politique supprimée et joindre la politique `AmazonECS_FullAccess`.

   a. Sur le Autorisations onglet, choisissez le X à côté du AmazonEC2ContainerServiceFullAccess politique.

   b. Choisissez Ajouter des autorisations.

   c. Choisissez Joindre directement les politiques existantes, recherchez et sélectionnez Amazon ECS_FullAccess politique, puis choisissez Suivant : Révision.

   d. Vérifiez vos modifications et choisissez Add permissions (Ajouter des autorisations).

   e. Répétez ces étapes pour chaque groupe, utilisateur ou rôle qui utilise la politique `AmazonEC2ContainerServiceFullAccess`.

Pour mettre à jour un groupe, un utilisateur ou un rôle afin d'utiliser **`Amazon ECS_FullAccess`** politique (AWS CLI)

1. Utilisez la commande generate-service-last-accessed-details pour générer un rapport qui inclut des détails sur la date à laquelle la politique supprimée a été utilisée pour la dernière fois.

   ```
   aws iam generate-service-last-accessed-details \
        --arn arn:aws:iam::aws:policy/AmazonEC2ContainerServiceFullAccess
   ```

   Exemple de sortie :

   ```
   {
       "JobId": "32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE"
   }
   ```

2. Utilisez l'ID de tâche indiqué dans la sortie précédente avec la commande get-service-last-accessed-details pour récupérer le dernier rapport consulté du service. Ce rapport affiche l'Amazon Resource Name (ARN) des entités IAM ayant utilisé la politique supprimée pour la dernière fois.

```
aws iam get-service-last-accessed-details \
      --job-id 32bb1fb0-1ee0-b08e-3626-ae83EXAMPLE
```

3. Utilisez l'une des commandes suivantes pour détacher la politique `AmazonEC2ContainerServiceFullAccess` d'un groupe, d'un utilisateur ou d'un rôle.

   - detach-group-policy
   - detach-role-policy
   - detach-user-policy

4. Utilisez l'une des commandes suivantes pour attacher la politique `AmazonECS_FullAccess` à un groupe, un utilisateur ou un rôle.

   - attach-group-policy
   - attach-role-policy
   - attach-user-policy

Utilisation des rôles liés à un service pour Amazon ECS

Utilisations d'Amazon Elastic Container Service Gestion des identités et des accès AWS (IAM) rôles liés aux services. Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon ECS. Service-linked les rôles sont prédéfinis par Amazon ECS et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Rubriques

- Utilisation de rôles pour permettre à Amazon ECS de gérer des clusters
- Utilisation de rôles pour gérer les instances gérées Amazon ECS

Utilisation de rôles pour permettre à Amazon ECS de gérer des clusters

Utilisations d'Amazon Elastic Container Service Gestion des identités et des accès AWS (IAM) rôles liés aux services. Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon ECS. Service-linked les rôles sont prédéfinis par Amazon ECS et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon ECS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon ECS définit les autorisations de ses rôles liés à

un service ; sauf définition contraire, seul Amazon ECS peut endosser ses 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.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon ECS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés aux services, consultez AWS services qui fonctionnent avec IAM et recherchez les services qui ont Oui dans le Service-linked rôles colonne. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Service-linked autorisations de rôle pour Amazon ECS

Amazon ECS utilise le rôle lié à un service nommé AWSServiceRoleForECS — Rôle permettant à Amazon ECS de gérer votre cluster.

Le AWSServiceRoleForECS le rôle lié à un service fait confiance aux services suivants pour assumer le rôle :

- `ecs.amazonaws.com`

La politique d'autorisations de rôle nommée AmazonECSServiceRolePolicy permet à Amazon ECS d'effectuer les actions suivantes sur les ressources spécifiées :

- Action : lorsque vous utilisez le mode réseau de `awsvpc` pour vos tâches Amazon ECS, Amazon ECS gère le cycle de vie des interfaces réseau élastiques associées à la tâche. Cela inclut également les balises que Amazon ECS ajoute à vos interfaces réseau élastiques.
- Action : lorsque vous utilisez un équilibreur de charge avec votre compte Amazon ECS, Amazon ECS gère l'enregistrement et l'annulation de l'enregistrement des ressources avec l'équilibreur de charge.
- Action : lorsque vous utilisez la découverte de service Amazon ECS, Amazon ECS gère la Route 53 requise et les ressources AWS Cloud Map pour que la découverte de services fonctionne.
- Action : lorsque vous utilisez le service de mise à l'échelle Amazon ECS, Amazon ECS gère les ressources Auto Scaling requises.
- Action : Amazon ECS crée et gère CloudWatch des alarmes et des flux de journaux qui vous aident à surveiller vos ressources Amazon ECS.

- Action : lorsque vous utilisez Amazon ECS Exec, Amazon ECS gère les autorisations nécessaires pour démarrer des sessions Amazon ECS Exec pour vos tâches.
- Action : lorsque vous utilisez Amazon ECS Service Connect, Amazon ECS gère les ressources AWS Cloud Map requises pour utiliser la fonctionnalité.
- Action : lorsque vous utilisez des fournisseurs de capacité Amazon ECS, Amazon ECS gère les autorisations requises pour modifier le groupe Auto Scaling et ses instances Amazon EC2.
- Action : Amazon ECS peut effectuer une mise à jour AWS Cloud Map attributs de service pour les services gérés par Amazon ECS.
- Action : Amazon ECS peut invoquer le provisionnement Amazon EC2 et désactiver l'ENI lors du démarrage et de l'arrêt de tâches.
- Action : Amazon ECS peut récupérer les fenêtres d'événements Amazon EC2 pour les services et les clusters associés aux fenêtres d'événements.

Vous devez configurer les autorisations de manière à permettre à vos utilisateurs, groupes ou rôles de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, voir Service-linked autorisations de rôle dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon ECS

Vous n’avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un cluster ou que vous créez ou mettez à jour un service dans Console de gestion AWS, le AWS CLI, ou le AWS API, Amazon ECS crée le rôle lié au service pour vous.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisiez le service Amazon ECS avant le 1er janvier 2017, date à laquelle il a commencé à prendre en charge les rôles liés aux services, Amazon ECS a créé le AWSServiceRoleForECS rôle dans votre compte. Pour en savoir plus, consultez Un nouveau rôle est apparu dans mon Compte AWS.

Vous pouvez également utiliser la console IAM pour créer un rôle lié à un service avec AWSServiceRoleForECS cas d'utilisation. Dans le AWS CLI ou le AWS API, utilisez IAM pour créer un rôle lié à un service avec `ecs.amazonaws.com` nom du 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.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un cluster, ou lorsque vous créez ou mettez à jour un service, Amazon ECS crée à nouveau le rôle lié au service pour vous.

Si vous supprimez ce rôle lié à un service, vous pouvez utiliser le même processus IAM pour recréer le rôle.

Modification d'un rôle lié à un service pour Amazon ECS

Amazon ECS ne vous permet pas de modifier AWSServiceRoleForECS rôle lié à un 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 plus d'informations, 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 Amazon ECS

Il n'est pas nécessaire de supprimer manuellement AWSServiceRoleForECS rôle. Lorsque vous supprimez des clusters dans toutes les régions du Console de gestion AWS, le AWS CLI, ou le AWS API, Amazon ECS nettoie les ressources et supprime le rôle lié au service pour vous.

Si vous n'avez plus besoin d'utiliser une fonctionnalité ou un service qui nécessite un rôle lié à un service, nous vous recommandons de supprimer ce rôle. De cette façon, vous n'avez aucune entité inutilisée qui n'est pas surveillée ou gérée activement. Cependant, vous devez nettoyer votre rôle lié à un service avant de pouvoir le supprimer manuellement.

Nettoyage d'un rôle lié à un service

Avant de pouvoir utiliser IAM pour supprimer un rôle lié à un service, vous devez supprimer toutes les ressources utilisées par le rôle.

> **Note**
>
> Si le service Amazon ECS utilise le rôle lorsque vous essayez de supprimer les ressources, la suppression peut échouer. Si cela se produit, patientez quelques minutes et réessayez.

Pour supprimer les ressources Amazon ECS utilisées par AWSServiceRoleForECS (console)

1. Arrêtez tous les services Amazon ECS en indiquant 0 comme nombre souhaité dans toutes les régions, puis supprimez-les. Pour plus d'informations, consultez Mettre à jour un service Amazon ECS et Suppression d'un service Amazon ECS à l'aide de la console.
2. Forcez l'annulation de l'enregistrement de toutes les instances de conteneur de tous les clusters dans toutes les régions. Pour plus d'informations, consultez Annulation de l'enregistrement d'instance de conteneur Amazon ECS.
3. Supprimez tous les clusters Amazon ECS dans toutes les régions. Pour plus d'informations, consultez Suppression d'un cluster Amazon ECS..

Pour supprimer les ressources Amazon ECS utilisées par AWSServiceRoleForECS (AWS CLI)

1. Arrêtez tous les services Amazon ECS en indiquant 0 comme nombre souhaité dans toutes les régions, puis supprimez-les. Pour plus d'informations, voir service de mise à jour et service de suppression dans le AWS Command Line Interface Référence.
2. Forcez l'annulation de l'enregistrement de toutes les instances de conteneur de tous les clusters dans toutes les régions. Pour plus d'informations, consultez la section deregister-container-instance.
3. Supprimez tous les clusters Amazon ECS dans toutes les régions. Pour en savoir plus, consultez la section delete-cluster.

Pour supprimer les ressources Amazon ECS utilisées par AWSServiceRoleForECS (API)

1. Arrêtez tous les services Amazon ECS en indiquant 0 comme nombre souhaité dans toutes les régions, puis supprimez-les. Pour plus d'informations, voir UpdateService et DeleteService dans le Référence de l'API Amazon ECS.
2. Forcez l'annulation de l'enregistrement de toutes les instances de conteneur de tous les clusters dans toutes les régions. Pour plus d'informations, consultez DeregisterContainerInstance.
3. Supprimez tous les clusters Amazon ECS dans toutes les régions. Pour plus d'informations, consultez DeleteCluster.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, AWS CLI, ou le AWS API pour supprimer le AWSServiceRoleForECS rôle lié à un service. Pour plus d'informations, consultez la section 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 Amazon ECS

Amazon ECS 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 la section Régions et points de terminaison AWS.

Utilisation de rôles pour gérer les instances gérées Amazon ECS

Utilisations d'Amazon Elastic Container Service Gestion des identités et des accès AWS (IAM) rôles liés aux services. Un rôle lié à un service est un type unique de rôle IAM directement lié à Amazon ECS. Service-linked les rôles sont prédéfinis par Amazon ECS et incluent toutes les autorisations dont le service a besoin pour appeler d'autres AWS services en votre nom.

Un rôle lié à un service simplifie la configuration d'Amazon ECS, car vous n'avez pas besoin d'ajouter manuellement les autorisations requises. Amazon ECS définit les autorisations de ses rôles liés à un service ; sauf définition contraire, seul Amazon ECS peut endosser ses 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.

Vous pouvez supprimer un rôle lié à un service uniquement après la suppression préalable de ses ressources connexes. Vos ressources Amazon ECS sont ainsi protégées, car vous ne pouvez pas involontairement supprimer l'autorisation d'accéder aux ressources.

Pour plus d'informations sur les autres services qui prennent en charge les rôles liés aux services, consultez AWS services qui fonctionnent avec IAM et recherchez les services qui ont Oui dans le Service-linked rôles colonne. Choisissez un Oui ayant un lien permettant de consulter les détails du rôle pour ce service.

Service-linked autorisations de rôle pour Amazon ECS

Amazon ECS utilise le rôle lié à un service nommé AWSServiceRoleForECSCompute — Rôle permettant à Amazon ECS de gérer les instances gérées Amazon EC2, mises en service par le fournisseur de capacité Amazon ECS Managed Instances.

Le AWSServiceRoleForECSCompute le rôle lié à un service fait confiance aux services suivants pour assumer le rôle :

- `ecs-compute.amazonaws.com`

La politique d'autorisations liée au rôle nommée `AmazonECSComputeServiceRolePolicy` permet à Amazon ECS de réaliser les actions suivantes :

- Amazon ECS peut décrire et supprimer des modèles de lancement.
- Amazon ECS peut décrire et supprimer les versions des modèles de lancement.
- Amazon ECS peut mettre fin à des instances.
- Amazon ECS peut décrire les paramètres de données d'instance suivants :
  - Instance
  - Interfaces réseau d'instance : Amazon ECS peut décrire les interfaces permettant de gérer le cycle de vie des instances EC2.
  - Fenêtre d'événements d'instance : Amazon ECS peut décrire les informations de la fenêtre d'événements afin de déterminer si le flux de travail peut être interrompu pour appliquer des correctifs à l'instance.
  - État de l'instance : Amazon ECS peut décrire l'état de l'instance afin de surveiller son état.

Vous devez configurer les autorisations de manière à permettre à vos utilisateurs, groupes ou rôles de créer, modifier ou supprimer un rôle lié à un service. Pour plus d'informations, voir Service-linked autorisations de rôle dans le Guide de l'utilisateur IAM.

Création d'un rôle lié à un service pour Amazon ECS

Vous n'avez pas besoin de créer manuellement un rôle lié à un service. Lorsque vous créez un fournisseur de capacité pour les instances gérées Amazon ECS dans Console de gestion AWS, le AWS CLI, ou le AWS API, Amazon ECS crée le rôle lié au service pour vous.

> **Important**
>
> Ce rôle lié à un service peut apparaître dans votre compte si vous avez effectué une action dans un autre service qui utilise les fonctions prises en charge par ce rôle. Si vous utilisiez le service Amazon ECS avant le 1er janvier 2017, date à laquelle il a commencé à prendre en charge les rôles liés aux services, Amazon ECS a créé le

AmazonECSComputeServiceRolePolicy rôle dans votre compte. Pour en savoir plus, consultez Un nouveau rôle est apparu dans mon Compte AWS.

Si vous supprimez ce rôle lié à un service et que vous avez ensuite besoin de le recréer, vous pouvez utiliser la même procédure pour recréer le rôle dans votre compte. Lorsque vous créez un fournisseur de capacité pour les instances gérées Amazon ECS, Amazon ECS recrée le rôle lié à un service en votre nom.

Si vous supprimez ce rôle lié à un service, vous pouvez utiliser le même processus IAM pour recréer le rôle.

Modification d'un rôle lié à un service pour Amazon ECS

Amazon ECS ne vous permet pas de modifier AmazonECSComputeServiceRolePolicy rôle lié à un 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 plus d'informations, 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 Amazon ECS

Vous n'avez pas besoin de supprimer manuellement le rôle AmazonECSComputeServiceRolePolicy . Lorsque vous supprimez tous les fournisseurs de capacité Amazon ECS Managed Instances dans toutes les régions du Console de gestion AWS, le AWS CLI, ou le AWS API, Amazon ECS nettoie les ressources et supprime le rôle lié au service pour vous.

Suppression manuelle du rôle lié au service

Utilisez la console IAM, AWS CLI, ou le AWS API pour supprimer le AmazonECSComputeServiceRolePolicy rôle lié à un service. Pour plus d'informations, consultez la section 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 Amazon ECS

Amazon ECS 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 la section Régions et points de terminaison AWS.

Rôles IAM pour Amazon ECS

Un rôle IAM est une identité IAM que vous pouvez créer dans votre compte et qui dispose d'autorisations spécifiques. Dans Amazon ECS, vous pouvez créer des rôles pour accorder des autorisations à des ressources Amazon ECS telles que des conteneurs ou des services.

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Utilisez le tableau suivant pour déterminer quels rôles IAM vous avez besoin pour Amazon ECS.

Role	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargate ou sur une instance externe et : • extrait une image de conteneur depuis un référentiel privé Amazon ECR. • extrait une image de conteneur à partir d'un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche ; • envoie les journaux des conteneurs à CloudWatch Journaux à l'aide du `awslogs` pilote de journaux.	Rôle IAM d'exécution de tâche Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
		Votre tâche est hébergée sur AWS Fargate ou des instances Amazon EC2 et : • utilise l’authentification de registre privé ; • utilise la surveillance d’exécution ; • la définition de tâche fait référence à des données sensibles à l'aide des secrets Secrets Manager ou AWS Paramètres du magasin de paramètres de Systems Manager.	
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS des services, tels qu'Amazon S3.	rôle IAM de tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos instances EC2 ou à vos instances externes de s’enregistrer auprès du cluster.	Votre tâche est hébergée sur des instances Amazon EC2 ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
Rôle Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'accéder AWS API.	Votre tâche est hébergée sur des instances externes.	Rôle IAM Amazon ECS Anywhere
Rôles d'infrastructure Amazon ECS pour les équilibreurs de charge	Ce rôle permet à Amazon ECS de gérer les ressources d'équilibrage de charge de vos clusters en votre nom pour blue/green déploiements.	Vous souhaitez utiliser Amazon ECS blue/green déploiements.	Rôle IAM d'infrastructure Amazon ECS pour les équilibreurs de charge
Amazon ECS CodeDeploy rôle	Ce rôle permet CodeDeploy pour mettre à jour vos services.	Vous utilisez le CodeDeploy blue/green type de déploiement pour déployer des services.	Amazon ECS CodeDeploy Rôle IAM
Amazon ECS EventBridge rôle	Ce rôle permet EventBridge pour mettre à jour vos services.	Vous utilisez le EventBridge règles et objectifs pour planifier vos tâches.	Amazon ECS EventBridge Rôle IAM

Role	Définition	En cas de besoin	En savoir plus
rôle d'infrastructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d'infrastructure de vos clusters.	• Vous souhaitez utiliser les instances gérées Amazon ECS pour vos capacités. • Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS dont le type de lancement est Fargate ou EC2. Le rôle d'infrast ructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM d'infrast ructure Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
Profil d’instance	Ce rôle permet aux instances gérées Amazon ECS d’assumer le rôle d’infrastructure en toute sécurité.	Vous utilisez des instances gérées Amazon ECS dans vos clusters.	Profil d’instance des instances gérées Amazon ECS

Pratiques exemplaires relatives aux rôles IAM dans Amazon ECS

Les rôles requis par Amazon ECS dépendent de la définition de tâche, du type de lancement et des fonctionnalités que vous utilisez. Nous vous recommandons de créer des rôles distincts dans le tableau plutôt que de partager les rôles.

Role	Définition	En cas de besoin	En savoir plus
Rôle d'exécution de tâche	Ce rôle permet à Amazon ECS d'utiliser d'autres AWS services en votre nom.	Votre tâche est hébergée sur AWS Fargate ou sur une instance externe et : • extrait une image de conteneur depuis un référentiel privé Amazon ECR. • extrait une image de conteneur à partir d’un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche ;	Rôle IAM d'exécution de tâche Amazon ECS

Role	Définition	En cas de besoin	En savoir plus
		• envoie les journaux des conteneur s à CloudWatch Journaux à l'aide du `awslogs` pilote de journaux. Votre tâche est hébergée sur AWS Fargate ou des instances Amazon EC2 et : • utilise l’authent ification de registre privé ; • utilise la surveilla nce d’exécution ; • la définition de tâche fait référence à des données sensibles à l'aide des secrets Secrets Manager ou AWS Paramètre s du magasin de paramètres de Systems Manager.	

Role	Définition	En cas de besoin	En savoir plus
Rôle de tâche	Ce rôle permet au code de votre application (sur le conteneur) d'utiliser d'autres AWS services.	Votre application accède à d'autres AWS des services, tels qu'Amazon S3.	rôle IAM de tâche Amazon ECS
Rôle de l'instance de conteneur	Ce rôle permet à vos instances EC2 ou à vos instances externes de s’enregistrer auprès du cluster.	Votre tâche est hébergée sur des instances Amazon EC2 ou sur une instance externe.	Rôle IAM d'instance de conteneur Amazon ECS
Rôle Amazon ECS Anywhere	Ce rôle permet à vos instances externes d'accéder AWS API.	Votre tâche est hébergée sur des instances externes.	Rôle IAM Amazon ECS Anywhere
Amazon ECS CodeDeploy rôle	Ce rôle permet CodeDeploy pour mettre à jour vos services.	Vous utilisez le CodeDeploy blue/green type de déploiement pour déployer des services.	Amazon ECS CodeDeploy Rôle IAM
Amazon ECS EventBridge rôle	Ce rôle permet EventBridge pour mettre à jour vos services.	Vous utilisez le EventBridge règles et objectifs pour planifier vos tâches.	Amazon ECS EventBridge Rôle IAM

Role	Définition	En cas de besoin	En savoir plus
rôle d’infrastructure Amazon ECS	Ce rôle permet à Amazon ECS de gérer les ressources d’infrastructure de vos clusters.	• Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS dont le type de lancement est Fargate ou EC2. Le rôle d’infrastructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches. • Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect. • Vous souhaitez créer des groupes cibles VPC Lattice.	Rôle IAM d’infrastructure Amazon ECS

Rôle de tâche

Nous vous recommandons d’attribuer un rôle de tâche. Son rôle peut être différencié du rôle de l'instance Amazon EC2 sur laquelle il s'exécute. 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.

Lorsque vous ajoutez un rôle de tâche à une définition de tâche, l'agent de conteneur Amazon ECS crée automatiquement un jeton avec un ID d'informations d'identification unique (par exemple, `12345678-90ab-cdef-1234-567890abcdef`) pour la tâche. Ce jeton et les informations d'identification du rôle sont ensuite ajoutés au cache interne de l'agent. L'agent remplit la variable d'environnement `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` du conteneur avec l'URI de l'ID d'informations d'identification (par exemple, `/v2/credentials/12345678-90ab-cdef-1234-567890abcdef`).

Vous pouvez récupérer manuellement les informations d'identification du rôle temporaire depuis un conteneur en ajoutant la variable d'environnement à l'adresse IP de l'agent de conteneur Amazon ECS et en exécutant la commande `curl` sur la chaîne résultante.

```
curl 169.254.170.2$AWS_CONTAINER_CREDENTIALS_RELATIVE_URI
```

La sortie attendue est la suivante :

```
{
 "RoleArn": "arn:aws:iam::123456789012:role/SSMTaskRole-SSMFargateTaskIAMRole-
DASWWSF2WGD6",
 "AccessKeyId": "AKIAIOSFODNN7EXAMPLE",
 "SecretAccessKey": "wJalrXUtnFEMI/K7MDENG/bPxRfiCYEXAMPLEKEY",
 "Token": "IQoJb3JpZ2luX2VjEEM/Example==",
 "Expiration": "2021-01-16T00:51:53Z"
}
```

Les nouvelles versions du AWS Les kits de développement logiciel récupèrent automatiquement ces informations d'identification à partir du `AWS_CONTAINER_CREDENTIALS_RELATIVE_URI` variable d'environnement lors de la création AWS Appels d'API. Pour plus d'informations sur le renouvellement des informations d'identification, voir Renouveler AWS informations d'identification sur RePost.

La sortie inclut une paire de clés d'accès composée d'un identifiant de clé d'accès secrète et d'une clé secrète que votre application utilise pour accéder AWS ressources. Il comprend également un jeton qui AWS permet de vérifier la validité des informations d'identification. Par défaut, les informations d'identification attribuées aux tâches utilisant des rôles de tâches sont valides pendant six heures. Ensuite, elles sont automatiquement soumises à une rotation par l'agent de conteneur Amazon ECS.

Rôle d'exécution de tâche

Le rôle d'exécution des tâches est utilisé pour accorder à l'agent de conteneur Amazon ECS l'autorisation d'appeler des appels spécifiques AWS Actions d'API en votre nom. Par exemple, lorsque vous utilisez AWS Fargate, Fargate a besoin d'un rôle IAM qui lui permet d'extraire des images d'Amazon ECR et d'écrire des journaux dans CloudWatch Journaux. Un rôle IAM est également requis lorsqu'une tâche fait référence à un secret stocké dans AWS Secrets Manager, par exemple un pull secret d'image.

> **Note**
>
> Si vous extrayez des images en tant qu'utilisateur authentifié, vous êtes moins susceptible d'être impacté par les modifications apportées aux limites de taux d'extraction de Docker Hub. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les instances de conteneur (langue française non garantie).
> En utilisant Amazon ECR et Amazon ECR Public, vous pouvez éviter les limites imposées par Docker. Si vous extrayez des images depuis Amazon ECR, cela permet également de raccourcir les temps d'extraction du réseau et de réduire les modifications de transfert de données lorsque le trafic quitte votre VPC.

> **Important**
>
> Lorsque vous utilisez Fargate, vous devez vous authentifier auprès d'un registre d'images privé à l'aide de `repositoryCredentials`. Il n'est pas possible de définir les variables d'environnement de l'agent de conteneur Amazon ECS `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA`, ou de modifier le fichier `ecs.config` pour les tâches hébergées sur Fargate. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Rôle de l'instance de conteneur

La stratégie IAM gérée par `AmazonEC2ContainerServiceforEC2Role` doit inclure les autorisations suivantes. Conformément aux conseils de sécurité standard pour accorder le moindre privilège, la stratégie gérée par `AmazonEC2ContainerServiceforEC2Role` peut être utilisée comme guide. Si vous n'avez pas besoin des autorisations accordées dans la stratégie gérée pour

votre cas d'utilisation, créez une stratégie personnalisée et ajoutez uniquement les autorisations dont vous avez besoin.

- `ec2:DescribeTags` : (facultatif) permet à un principal de décrire les balises associées à une instance Amazon EC2. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour prendre en charge la propagation des balises de ressources. Pour plus d'informations, consultez Comment les ressources sont étiquetées.
- `ecs:CreateCluster` : (facultatif) permet au principal de créer un cluster Amazon ECS. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour créer un cluster `default`, s'il n'existe pas déjà.
- `ecs:DeregisterContainerInstance` : (facultatif) permet au principal d'annuler l'enregistrement d'une instance de conteneur Amazon ECS auprès d'un cluster. L'agent de conteneur Amazon ECS n'appelle pas cette opération d'API, mais cette autorisation est conservée afin de garantir la compatibilité ascendante.
- `ecs:DiscoverPollEndpoint` : (requis) cette action renvoie les points de terminaison utilisés par l'agent de conteneur Amazon ECS pour rechercher les mises à jour.
- `ecs:Poll` : (requis) permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS pour signaler les changements d'état de tâche.
- `ecs:RegisterContainerInstance` : (requis) permet au principal d'enregistrer une instance de conteneur auprès d'un cluster. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour enregistrer l'instance Amazon EC2 avec un cluster ainsi que pour prendre en charge la propagation des balises de ressources.
- `ecs:StartTelemetrySession` : (facultatif) permet à l'agent de conteneur Amazon ECS de communiquer avec le plan de contrôle Amazon ECS pour signaler les informations et métriques d'état de chaque conteneur et tâche.

  Bien que cette autorisation ne soit pas obligatoire, nous vous recommandons de l'ajouter pour permettre aux métriques de l'instance de conteneur de démarrer des actions de mise à l'échelle et de recevoir des rapports relatifs aux commandes de surveillance de l'état.
- `ecs:TagResource` : (facultatif) permet à l'agent de conteneur Amazon ECS d'étiqueter le cluster lors de sa création et d'étiqueter les instances de conteneur lorsqu'elles sont enregistrées dans un cluster.
- `ecs:UpdateContainerInstancesState` : permet au principal de modifier le statut d'une instance de conteneur Amazon ECS. Cette autorisation est utilisée par l'agent de conteneur Amazon ECS pour le drainage des instances Spot.

- `ecs:Submit*` : (requis) cela inclut les actions d'API `SubmitAttachmentStateChanges`, `SubmitContainerStateChange` et `SubmitTaskStateChange`. Elles sont utilisées par l'agent de conteneur Amazon ECS pour signaler les modifications d'état pour chaque ressource au plan de contrôle Amazon ECS. L'autorisation `SubmitContainerStateChange` n'est plus utilisée par l'agent de conteneur Amazon ECS, mais est conservée afin de garantir la compatibilité ascendante.
- `ecr:GetAuthorizationToken` : (facultatif) permet au principal de récupérer un jeton d'autorisation. Un jeton d'autorisation représente vos informations d'authentification IAM et peut être utilisé pour accéder à n'importe quel registre Amazon ECR auquel votre principal IAM a accès. Le jeton d'autorisation reçu est valide pendant 12 heures.
- `ecr:BatchCheckLayerAvailability` : (facultatif) lorsqu'une image de conteneur est transférée vers un référentiel privé Amazon ECR, chaque couche d'image est examinée afin de vérifier si elle a déjà été transférée. Le cas échéant, le calque d'image est ignoré.
- `ecr:GetDownloadUrlForLayer` : (facultatif) lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une fois pour chaque couche d'image qui n'est pas déjà mise en cache.
- `ecr:BatchGetImage` : (facultatif) lorsqu'une image de conteneur est extraite d'un référentiel privé Amazon ECR, cette API est appelée une fois pour récupérer le manifeste d'image.
- `logs:CreateLogStream` — (Facultatif) Permet à un directeur de créer un CloudWatch Enregistre le flux de journaux pour un groupe de journaux spécifié.
- `logs:PutLogEvents` : (facultatif) permet au principal de charger un lot d'événements de journaux dans un flux de journaux spécifié.

Service-linked rôles

Vous pouvez utiliser le rôle lié à un service pour Amazon ECS pour octroyer au service Amazon ECS l'autorisation d'appeler d'autres API de service en votre nom. Amazon ECS a besoin des autorisations pour créer et supprimer des interfaces réseau, enregistrer et annuler l'enregistrement des cibles auprès d'un groupe cible. Il doit également disposer des autorisations nécessaires pour créer et supprimer des stratégies de mise à l'échelle. Ces autorisations sont octroyées par le biais d'un rôle lié à un service. Ce rôle est créé en votre nom la première fois que vous utilisez le service.

> **Note**
>
> Si vous supprimez par inadvertance le rôle lié à un service, vous pouvez le recréer. Pour obtenir des instructions, veuillez consulter Création du rôle lié à un service.

Recommandations relatives aux rôles

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez les rôles et politiques IAM de votre tâche.

Bloquez l'accès aux métadonnées Amazon EC2

Lorsque vous exécutez vos tâches sur des instances Amazon EC2, nous vous recommandons vivement de bloquer l'accès des conteneurs au service de métadonnées d'instance Amazon EC2 (IMDS) afin d'empêcher vos conteneurs d'accéder à des données sensibles au niveau de l'instance, y compris les informations d'identification du profil d'instance de l'instance. Si vos applications doivent appeler un AWS Action d'API, utilisez plutôt les rôles IAM pour les tâches.

Pour empêcher les tâches exécutées en mode pont d'accéder aux métadonnées Amazon EC2, exécutez la commande suivante ou mettez à jour les données utilisateur de l'instance. Pour plus d'instructions sur la mise à jour des données utilisateur d'une instance, veuillez consulter cet article d'AWS Support. Pour plus d'informations sur le mode pont de définition des tâches, veuillez consulter Mode réseau de définition des tâches (langue française non garantie).

```
sudo yum install -y iptables-services; sudo iptables --insert FORWARD 1 --in-interface
 docker+ --destination 169.254.169.254/32 --jump DROP
```

Pour que cette modification soit maintenue après un redémarrage, exécutez la commande suivante, spécifique à votre Amazon Machine Image (AMI) :

- Amazon Linux 2

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now
   iptables
  ```

- Amazon Linux

  ```
  sudo service iptables save
  ```

Pour les tâches qui utilisent le mode réseau `awsvpc`, définissez la variable d'environnement `ECS_AWSVPC_BLOCK_IMDS` sur `true` dans le fichier `/etc/ecs/ecs.config`.

Vous devez définir la variable `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` sur `false` dans le fichier `ecs-agent config` pour empêcher les conteneurs exécutés sur le réseau `host` d'accéder aux métadonnées Amazon EC2.

Utilisez le **`awsvpc`** mode réseau

Utilisez le mode réseau `awsvpc` pour limiter le flux de trafic entre les différentes tâches ou entre vos tâches et les autres services exécutés au sein de votre Amazon VPC. Cela ajoute une couche de sécurité supplémentaire. Le mode réseau `awsvpc` fournit une isolation réseau au niveau des tâches pour les tâches exécutées sur Amazon EC2. Il s'agit du mode par défaut activé AWS Fargate. c'est le seul mode réseau que vous pouvez utiliser pour attribuer un groupe de sécurité à des tâches.

Utilisation des dernières informations consultées pour affiner les rôles

Nous vous recommandons de supprimer toutes les actions qui n'ont jamais été utilisées ou qui n'ont pas été utilisées depuis un certain temps. Cela empêche tout accès indésirable. Pour ce faire, examinez les dernières informations fournies par IAM, puis supprimez les actions qui n’ont jamais été utilisées ou qui n’ont pas été utilisées récemment. Les étapes de la section suivantes décrivent comment procéder.

Utilisez la commande suivante pour générer un rapport contenant les dernières informations d'accès pour la stratégie référencée :

```
aws iam generate-service-last-accessed-details --arn arn:aws:iam::123456789012:policy/
ExamplePolicy1
```

Utilisez le `JobId` de la sortie pour exécuter la commande suivante. Après cela, vous pouvez consulter les résultats du rapport.

```
aws iam get-service-last-accessed-details --job-id 98a765b4-3cde-2101-2345-example678f9
```

Pour plus d'informations, voir Affinez les autorisations dans AWS en utilisant les dernières informations consultées.

Surveillance AWS CloudTrail pour activité suspecte

Vous pouvez surveiller AWS CloudTrail pour toute activité suspecte. La plupart AWS Les appels d'API sont enregistrés dans AWS CloudTrail en tant qu'événements. Ils sont analysés par AWS

CloudTrail Insights, et vous êtes alerté de tout comportement suspect associé à `write` Appels d'API. Cela peut inclure un pic du volume d'appels. Ces alertes incluent des informations telles que l'heure à laquelle l'activité inhabituelle s'est produite et le principal ARN d'identité ayant contribué aux API.

Vous pouvez identifier les actions effectuées par les tâches ayant un rôle IAM dans AWS CloudTrail examinant la propriété `userIdentity` de l'événement. Dans l'exemple suivant, l'`arn` comprend le nom du rôle endossé, `s3-write-go-bucket-role`, suivi du nom de la tâche, `7e9894e088ad416eb5cab92afExample`.

```
"userIdentity": {
    "type": "AssumedRole",
    "principalId": "AROA36C6WWEJ2YEXAMPLE:7e9894e088ad416eb5cab92afExample",
    "arn": "arn:aws:sts::123456789012:assumed-role/s3-write-go-bucket-
role/7e9894e088ad416eb5cab92afExample",
    ...
}
```

> **Note**
>
> Lorsque des tâches qui endossent un rôle sont exécutées sur des instances de conteneur Amazon EC2, une demande est journalisée par l'agent de conteneur Amazon ECS dans le journal d'audit de l'agent situé à une adresse au format `/var/log/ecs/audit.log.YYYY-MM-DD-HH`. Pour plus d'informations, veuillez consulter Journal des rôles IAM pour les tâches et Journalisation des événements Insights pour les journaux de suivi.

Rôle IAM d'exécution de tâche Amazon ECS

Le rôle IAM d'exécution de tâche accorde aux agents de conteneur Amazon ECS et Fargate l'autorisation d'effectuer des appels d'API AWS en votre nom. Le rôle IAM d'exécution de la tâche est requis en fonction des besoins de votre tâche. Vous pouvez avoir plusieurs rôles d'exécution de tâche pour différents objectifs et services associés à votre compte.

> **Note**
>
> Ces autorisations sont mises à la disposition de l’agent exécuté sur votre instance par Amazon ECS qui lui envoie régulièrement les informations d’identification temporaires du rôle, mais elles ne sont pas directement accessibles par les conteneurs de la tâche. Pour

connaître les autorisations IAM dont votre code d'application à l'intérieur du conteneur a besoin pour s'exécuter, consultez la section rôle IAM de tâche Amazon ECS.

Voici des cas d'utilisation courants pour un rôle IAM d'exécution de tâche :

- Votre tâche est hébergée sur AWS Fargate, des instances gérées Amazon ECS ou une instance externe et :
  - extrait une image de conteneur depuis un référentiel privé Amazon ECR.
  - extrait une image de conteneur à partir d'un référentiel privé Amazon ECR dans un compte différent de celui qui exécute la tâche ;
  - envoie les journaux des conteneurs à CloudWatch Journaux à l'aide du `awslogs` pilote de journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à CloudWatch.
- Vos tâches sont hébergées soit sur AWS Fargate ou des instances Amazon EC2 et :
  - utilise l'authentification de registre privé ; Pour plus d'informations, consultez Autorisations d'authentification du registre privé.
  - utilise la surveillance d'exécution ;
  - la définition de la tâche fait référence à des données sensibles à l'aide des secrets Secrets Manager ou des paramètres du magasin de paramètres AWS Systems Manager. Pour plus d'informations, consultez Autorisations Secrets Manager ou Systems Manager.

> **Note**
>
> Le rôle d'exécution de tâche est pris en charge par l'agent de conteneur Amazon ECS, version 1.16.0 et ultérieures.

Amazon ECS fournit la politique gérée nommée `AmazonECSTaskExecutionRolePolicy` qui contient les autorisations requises pour les cas d'utilisation courants décrits précédemment. Pour plus d'informations, voir AmazonECSTaskExecutionRolePolicy dans le AWS Guide de référence des politiques gérées. Il peut être nécessaire d'ajouter des politiques intégrées à votre rôle d'exécution des tâches pour des cas d'utilisation particuliers.

La console Amazon ECS crée un rôle d'exécution des tâches. Vous pouvez associer manuellement la politique IAM gérée pour les tâches afin de permettre à Amazon ECS d'ajouter des autorisations pour les fonctionnalités et améliorations futures à mesure qu'elles sont introduites. Vous pouvez

utiliser la recherche sur console IAM pour rechercher `ecsTaskExecutionRole` et voir si votre compte possède déjà le rôle d'exécution des tâches. Pour plus d'informations, consultez la section Recherche dans la console IAM dans le Guide de l'utilisateur IAM.

Si vous téléchargez des images en tant qu'utilisateur authentifié, vous serez moins susceptible d'être affecté par les changements apportés à l'utilisation et aux limites de Docker Hub. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les instances de conteneur (langue française non garantie).

En utilisant Amazon ECR et Amazon ECR Public, vous pouvez éviter les limites imposées par Docker. Si vous extrayez des images depuis Amazon ECR, cela permet également de raccourcir les temps d'extraction du réseau et de réduire les modifications de transfert de données lorsque le trafic quitte votre VPC.

Lorsque vous utilisez Fargate, vous devez vous authentifier auprès d'un registre d'images privé à l'aide de `repositoryCredentials`. Il n'est pas possible de définir les variables d'environnement de l'agent de conteneur Amazon ECS `ECS_ENGINE_AUTH_TYPE` ou `ECS_ENGINE_AUTH_DATA`, ou de modifier le fichier `ecs.config` pour les tâches hébergées sur Fargate. Pour plus d'informations, veuillez consulter Authentification de registre privé pour les tâches (langue française non garantie).

Création du rôle d'exécution de tâche

Si votre compte ne possède pas encore de rôle d'exécution de tâche, procédez comme suit pour créer le rôle.

Console de gestion AWS

Pour créer un rôle de service Elastic Container Service (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Rôles, puis Créer un rôle.
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation Tâche Elastic Container Service.
5. Choisissez Suivant.
6. Dans le Ajouter des autorisations section, recherchez AmazonECSTaskExecutionRolePolicy, puis sélectionnez la politique.
7. Choisissez Suivant.

8. Pour Nom du rôle, entrez ecsTaskExecutionRole.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tout *user input* avec vos propres informations.

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs-tasks.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Créez un rôle IAM nommé `ecsTaskExecutionRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsTaskExecutionRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Joignez le AWS géré `AmazonECSTaskExecutionRolePolicy` politique à l'égard de `ecsTaskExecutionRole` rôle.

```
aws iam attach-role-policy \
      --role-name ecsTaskExecutionRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSTaskExecutionRolePolicy
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Extraire des images de conteneurs à partir de registres privés extérieurs à AWS (tels que Docker Hub, Quay.io, ou votre propre registre privé) à l'aide des informations d'identification de Secrets Manager	Autorisations d’authentification du registre privé
Transmettre des données sensibles avec Systems Manager ou Secrets Manager	Autorisations Secrets Manager ou Systems Manager
Faire en sorte que les tâches Fargate extraient les images Amazon ECR via les points de terminaison de l’interface	Tâches Fargate extrayant des images Amazon ECR via les autorisations des points de terminaison d’interface
Héberger les fichiers de configuration dans un compartiment Amazon S3	Autorisations de stockage de fichiers Amazon S3
Configurer Container Insights pour afficher les événements du cycle de vie Amazon ECS	Autorisations requises pour l’affichage des événements du cycle de vie Amazon ECS dans Container Insights
Afficher les événements du cycle de vie d’Amazon ECS dans Container Insights	Autorisations requises pour afficher les événements du cycle de vie Amazon ECS dans Container Insights

Autorisations d’authentification du registre privé

L'authentification de registre privé permet à vos tâches Amazon ECS d'extraire des images de conteneurs depuis des registres privés extérieurs à AWS (tels que Docker Hub, Quay.io, ou votre propre registre privé) qui nécessitent des informations d'authentification. Cette fonctionnalité utilise Secrets Manager pour stocker en toute sécurité vos informations d’identification de registre, qui sont ensuite référencées dans votre définition de tâche à l’aide du paramètre `repositoryCredentials`.

Pour plus d'informations sur la configuration de l'authentification dans le registre privé, voir En utilisant des produits non AWS images de conteneurs dans Amazon ECS.

Pour donner accès aux secrets contenant vos informations d'identification du registre privé, ajoutez les autorisations suivantes en tant que politique intégrée au rôle d'exécution des tâches. Pour plus d'informations, consultez Ajout et suppression de politiques IAM.

- `secretsmanager:GetSecretValue` : nécessaire pour récupérer les informations d'identification du registre privé auprès de Secrets Manager.
- `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé KMS personnalisée et non la clé par défaut. L'Amazon Resource Name (ARN) de votre clé personnalisée doit être ajouté en tant que ressource.

Voici un exemple de stratégie en ligne qui ajoute les autorisations.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "kms:Decrypt",
                "secretsmanager:GetSecretValue"
            ],
            "Resource": [
                "arn:aws:secretsmanager:us-
east-1:111122223333:secret:secret_name",
                "arn:aws:kms:us-east-1:111122223333:key/key_id"
            ]
        }
    ]
}
```

Autorisations Secrets Manager ou Systems Manager

L'autorisation d'autoriser l'agent du conteneur à extraire le nécessaire AWS Systems Manager ou des ressources sur Secrets Manager. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.

Utilisation de Secrets Manager

Pour fournir l'accès aux secrets Secrets Manager que vous créez, ajoutez manuellement l'autorisation suivante au rôle d'exécution de tâche. Pour plus d'informations sur la gestion des autorisations, consultez Ajout et suppression de stratégies IAM dans le Guide de l'utilisateur IAM.

- `secretsmanager:GetSecretValue` : obligatoire si vous faites référence à un secret Secrets Manager. Ajoute l'autorisation pour récupérer le secret depuis Secrets Manager.

L'exemple suivant de politique ajoute les autorisations requises.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "secretsmanager:GetSecretValue"
      ],
      "Resource": [
        "arn:aws:secretsmanager:us-east-1:111122223333:secret:secret_name"
      ]
    }
  ]
}
```

Utilisation de Systems Manager

> **Important**
>
> Pour les tâches qui utilisent le type de lancement EC2, vous devez utiliser la variable de configuration de l'agent ECS `ECS_ENABLE_AWSLOGS_EXECUTIONROLE_OVERRIDE=true`

pour utiliser cette fonction. Vous pouvez l'ajouter au fichier `./etc/ecs/ecs.config` lors de la création de l'instance de conteneur, ou vous pouvez l'ajouter à une instance existante, puis redémarrer l'agent ECS. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Pour permettre l'accès aux paramètres du magasin de paramètres du gestionnaire de systèmes que vous créez, ajoutez manuellement les autorisations suivantes en tant que stratégie au rôle d'exécution de tâche. Pour plus d'informations sur la gestion des autorisations, consultez Ajout et suppression de stratégies IAM dans le Guide de l'utilisateur IAM.

- `ssm:GetParameters` : obligatoire lorsque vous référencez un paramètre Systems Manager Parameter Store dans une définition de tâche. Ajoute l'autorisation pour récupérer les paramètres de Systems Manager.
- `secretsmanager:GetSecretValue` : obligatoire si vous référencez directement un secret Secrets Manager ou si votre paramètre Systems Manager Parameter Store référence un secret Secrets Manager dans une définition de tâche. Ajoute l'autorisation pour récupérer le secret depuis Secrets Manager.
- `kms:Decrypt` : obligatoire uniquement si votre secret utilise une clé managée client et non la clé par défaut. L'ARN de votre clé personnalisée doit être ajouté en tant que ressource. Ajoute l'autorisation pour déchiffrer la clé gérée par le client.

L'exemple suivant de politique ajoute les autorisations requises.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ssm:GetParameters",
        "secretsmanager:GetSecretValue",
        "kms:Decrypt"
      ],
      "Resource": [
        "arn:aws:ssm:us-east-1:111122223333:parameter/parameter_name",
```

```
        "arn:aws:secretsmanager:us-east-1:111122223333:secret:secret_name",
        "arn:aws:kms:us-east-1:111122223333:key/key_id"
      ]
    }
  ]
}
```

Tâches Fargate extrayant des images Amazon ECR via les autorisations des points de terminaison d'interface

Lors du lancement des tâches qui utilisent Fargate et qui extraient des images à partir d'Amazon ECR, lorsqu'Amazon ECR est configuré pour utiliser un point de terminaison de VPC d'interface, vous pouvez restreindre les droits d'accès des tâches à un VPC ou à un point de terminaison de VPC spécifique. Pour ce faire, créez un rôle d'exécution de tâche pour les tâches qui utilisent des clés de condition IAM.

Utilisez les clés de condition globales IAM suivantes pour restreindre l'accès à un VPC ou à un point de terminaison d'un VPC spécifique. Pour de plus amples informations, veuillez consulter Clés de contexte de condition globale AWS.

- `aws:SourceVpc` : restreint l'accès à un VPC spécifique. Vous pouvez limiter le VPC au VPC qui héberge la tâche et le point de terminaison.
- `aws:SourceVpce` : restreint l'accès à un point de terminaison d'un VPC spécifique.

La stratégie de rôle d'exécution de tâche suivante fournit un exemple d'ajout de clés de condition :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
            {
                "Effect": "Allow",
                "Action": [
                    "ecr:GetAuthorizationToken",
                    "logs:CreateLogStream",
                    "logs:PutLogEvents"
                ],
                "Resource": "*"
```

```
            },
            {
                "Effect": "Allow",
                "Action": [
                    "ecr:BatchCheckLayerAvailability",
                    "ecr:GetDownloadUrlForLayer",
                    "ecr:BatchGetImage"
                ],
                "Resource": "arn:aws:ecr:*:*:repository/*",
                "Condition": {
                    "StringEquals": {
                            "aws:sourceVpce": "vpce-0123456789abcdef0"
                    }
                }
            }
    ]
}
```

Autorisations Amazon ECR

Les autorisations suivantes sont requises lorsque vous devez extraire des images de conteneur à partir des référentiels privés Amazon ECR. Le rôle d'exécution de tâche doit disposer de ces autorisations pour permettre au conteneur Amazon ECS et aux agents Fargate d'extraire les images du conteneur en votre nom. Pour les implémentations ECS de base, ces autorisations doivent être ajoutées au rôle d'exécution des tâches plutôt qu'au rôle IAM de tâche.

La politique gérée du rôle d'exécution des tâches Amazon ECS (`AmazonECSTaskExecutionRolePolicy`) inclut les autorisations nécessaires pour extraire des images depuis Amazon ECR. Si vous utilisez la politique gérée, vous n'avez pas besoin d'ajouter ces autorisations séparément.

Si vous créez une politique personnalisée, incluez les autorisations suivantes pour autoriser l'extraction d'images depuis Amazon ECR :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
```

```
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Notez que ces autorisations sont différentes de celles qui peuvent être requises dans le rôle IAM de tâche si le code de votre application doit interagir directement avec les API Amazon ECR. Pour plus d'informations sur les autorisations du rôle IAM de tâche pour Amazon ECR, consultez la section Autorisations Amazon ECR.

Autorisations de stockage de fichiers Amazon S3

Lorsque vous spécifiez un fichier de configuration hébergé dans Amazon S3, le rôle d'exécution de la tâche doit inclure l'autorisation `s3:GetObject` pour le fichier de configuration et l'autorisation `s3:GetBucketLocation` sur le compartiment Amazon S3 dans lequel se trouve le fichier. Pour plus d'informations, consultez la section Actions de politiques pour Amazon S3 dans le Guide de l'utilisateur Amazon Simple Storage Service.

L'exemple de politique suivant ajoute les autorisations requises pour récupérer un fichier à partir d'Amazon S3. Spécifiez le nom de votre compartiment Amazon S3 et le nom du fichier de configuration.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetObject"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket/folder_name/config_file_name"
```

```
      ]
    },
    {
      "Effect": "Allow",
      "Action": [
        "s3:GetBucketLocation"
      ],
      "Resource": [
        "arn:aws:s3:::amzn-s3-demo-bucket"
      ]
    }
  ]
}
```

Considération importante en matière de sécurité

Lorsque vous utilisez les fonctionnalités d'Amazon ECS qui s'intègrent aux compartiments Amazon S3, implémentez la validation de la propriété des compartiments pour empêcher les attaques par prise de contrôle des compartiments. Sans validation appropriée, si un compartiment Amazon S3 est supprimé et recréé par un acteur malveillant portant le même nom, vos tâches peuvent charger à votre insu des configurations malveillantes ou envoyer des données sensibles vers des compartiments contrôlés par les attaquants.

Condition de politique IAM recommandée :

```
"Condition": {
  "StringEquals": {
    "aws:ResourceAccount": "TRUSTED-ACCOUNT-ID"
  }
}
```

Remplacer *TRUSTED-ACCOUNT-ID* avec le AWS ID de compte propriétaire du compartiment S3.

Cette condition garantit que votre rôle d'exécution des tâches ne peut accéder qu'aux compartiments Amazon S3 appartenant au compte sécurisé spécifié.

rôle IAM de tâche Amazon ECS

Vos tâches Amazon ECS peuvent être associées à un rôle IAM. Les autorisations octroyées dans le rôle IAM sont transmises aux conteneurs exécutés dans la tâche. Ce rôle permet au code de votre application (exécutée dans le conteneur) d'utiliser d'autres AWS services. Le rôle de tâche est requis lorsque votre application accède à d'autres AWS des services, tels qu'Amazon S3.

> **Note**
>
> Le conteneur Amazon ECS et les agents Fargate n'accèdent pas à ces autorisations. Pour connaître les autorisations IAM dont Amazon ECS a besoin pour extraire des images de conteneurs et exécuter la tâche, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

Voici les avantages liés à l'utilisation des rôles de tâche :

- Séparation des préoccupations : si vous utilisez EC2, les rôles IAM de tâche vous permettent de spécifier des autorisations IAM pour vos conteneurs sans avoir à spécifier ces autorisations à l'aide des profils d'instance EC2 (pour plus d'informations, consultez la section Utilisation des profils d'instance dans le Guide de l'utilisateur Gestion des identités et des accès AWS). Par conséquent, vous pouvez déployer vos applications de manière indépendante et uniforme sur les instances de conteneur ECS sans avoir à modifier les autorisations IAM associées aux instances EC2.
- Auditabilité: L'accès et la journalisation des événements sont disponibles via CloudTrail pour garantir un audit rétrospectif. Les informations d'identification des tâches ont pour contexte «`taskArn`'qui est joint à la session, donc CloudTrail les journaux indiquent pour quelle tâche les informations d'identification du rôle ont été envoyées.
- Fourniture d'informations d'identification uniforme : ECS fournit des informations d'identification de rôle IAM à vos conteneurs et les rend accessibles via une interface bien définie, quelle que soit l'option de calcul associée à vos tâches. Sur ECS Fargate, les profils d'instance EC2 ne sont pas disponibles pour les conteneurs de vos tâches. Les rôles IAM des tâches vous permettent d'associer des autorisations IAM à vos conteneurs, quelle que soit l'option de calcul que vous utilisez AWS SDK ou AWS CLI dans vos conteneurs. Pour plus d'informations sur la façon dont AWS Le SDK accède à ces informations d'identification, voir Fournisseur d'informations d'identification de conteneurs.

Important

Les conteneurs ne constituent pas une limite de sécurité et l'utilisation de rôles IAM de tâche ne change rien à cela. Chaque tâche s'exécutant sur Fargate possède sa propre limite d'isolement et ne partage pas le noyau sous-jacent, les ressources d'UC, les ressources de mémoire ou l'interface réseau Elastic avec une autre tâche. Pour les charges de travail soumises à des exigences d'isolation strictes, utilisez Fargate. Pour les instances de conteneur EC2, ECS Managed Instances et ECS Anywhere, il n'existe aucune isolation des tâches et les conteneurs peuvent potentiellement accéder aux informations d'identification pour d'autres tâches sur la même instance de conteneur. Ils peuvent également accéder aux autorisations attribuées au Rôle d'instance de conteneur ECS et d'autres données sensibles telles que les métadonnées d'instance et les informations d'identification des rôles IAM accessibles via le service de métadonnées d'instance Amazon EC2 (IMDS), ainsi que les données provenant de tâches colocalisées. Suivez les recommandations figurant dans Recommandations relatives aux rôles pour empêcher les conteneurs d'accéder au service de métadonnées d'instance Amazon EC2 sur vos instances de conteneurs (pour plus d'informations, consultez Utiliser le service de métadonnées d'instance pour accéder aux métadonnées d'instance dans le Guide de l'utilisateur d'Amazon EC2).
Notez que lorsque vous spécifiez un rôle IAM pour une tâche, AWS La CLI ou les autres kits de développement logiciel contenus dans les conteneurs pour cette tâche utilisent le AWS informations d'identification fournies exclusivement par le rôle de tâche et ils n'héritent d'aucune autorisation IAM de l'instance Amazon EC2 ou externe sur laquelle ils s'exécutent.

Création du rôle IAM de tâche

Lorsque vous créez une politique IAM à utiliser pour vos tâches, celle-ci doit inclure les autorisations que vous souhaitez attribuer aux conteneurs de vos tâches. Vous pouvez utiliser un AWS politique gérée, ou vous pouvez créer une politique personnalisée à partir de zéro qui répond à vos besoins spécifiques. Pour plus d'informations, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Important

Pour les tâches Amazon ECS (pour tous les types de lancement), nous vous recommandons d'utiliser la politique et le rôle IAM pour vos tâches. Ces informations d'identification permettent à votre tâche d'effectuer AWS Requêtes d'API sans appel `sts:AssumeRole` pour assumer le même rôle que celui qui est déjà associé à la tâche. Si votre tâche nécessite

qu'un rôle s'assume, vous devez créer une politique de confiance qui autorise explicitement ce rôle à s'assumer. Pour de plus amples informations, consultez la section Mise à jour d'une stratégie d'approbation de rôle dans le Guide de l'utilisateur IAM.

Après la création de la politique IAM, vous pouvez créer un rôle IAM qui inclut la politique que vous référencez dans votre définition de tâche Amazon ECS. Vous pouvez créer le rôle à l'aide du cas d'utilisation Amazon Elastic Container Service Task dans la console IAM. Vous pouvez ensuite lier votre politique IAM spécifique au rôle qui attribue aux conteneurs de votre tâche les autorisations que vous souhaitez. Les procédures suivantes décrivent la marche à suivre.

Si vous avez plusieurs définitions de tâche ou services qui requièrent des autorisations IAM, nous vous conseillons de créer un rôle pour chaque définition de tâche ou service en lui affectant les autorisations minimales requises pour les tâches à exécuter, afin de minimiser les accès octroyés à chaque tâche.

Pour plus d'informations sur le point de terminaison de service de votre région, consultez la section Points de terminaison de service dans le Guide Référence générale d'Amazon Web Services .

Le rôle de tâche IAM doit posséder une politique de confiance qui spécifie le service `ecs-tasks.amazonaws.com`. L'autorisation `sts:AssumeRole` permet à vos tâches de prendre un rôle IAM différent de celui utilisé par votre instance Amazon EC2. De cette façon, votre tâche n'hérite pas du rôle associé à l'instance Amazon EC2. Voici un exemple de politique de confiance. Remplacez l'identifiant de région et spécifiez AWS numéro de compte que vous utilisez pour lancer des tâches.

Important

Lorsque vous créez le rôle IAM de votre tâche, nous vous recommandons d'utiliser le `aws:SourceAccount` ou `aws:SourceArn` des clés de condition dans la politique de relation de confiance associée au rôle pour étendre davantage les autorisations afin d'éviter tout problème de sécurité confus lié à l'adjoint. L'utilisation de la clé de condition `aws:SourceArn` pour spécifier un cluster spécifique n'est actuellement pas prise en charge, vous devez utiliser le caractère générique pour spécifier tous les clusters. Pour en savoir plus sur le problème des adjoints confus et sur la manière de protéger votre AWS compte, voir Le problème des députés confus dans le Guide de l'utilisateur IAM.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
   ]
}
```

Utilisez la procédure suivante pour créer une politique de récupération d'objets depuis Amazon S3 avec un exemple de stratégie. Remplacez tous les *user input* par vos propres valeurs.

Console de gestion AWS

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
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 Mise en route.
3. En haut de la page, sélectionnez Créer une politique.

4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ]
      }
   ]
}
```

6. 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 plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez 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.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

AWS CLI

Remplacez tous les *user input* par vos propres valeurs.

1. Créez un fichier nommé `s3-policy.json` avec le contenu suivant.

JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Action":[
            "s3:GetObject"
         ],
         "Resource":[
            "arn:aws:s3:::my-task-secrets-bucket/*"
         ]
      }
   ]
}
```

2. Utilisez la commande suivante pour créer la politique IAM à l'aide du fichier de politique JSON. Remplacez tous les *user input* par vos propres valeurs.

```
aws iam create-policy \
      --policy-name taskRolePolicy \
      --policy-document file://s3-policy.json
```

Utilisez la procédure suivante pour créer le rôle de service.

Console de gestion AWS

Pour créer un rôle de service Elastic Container Service (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Rôles, puis Créer un rôle.
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation Tâche Elastic Container Service.
5. Choisissez Suivant.

6. Pour Attacher des autorisations, recherchez et sélectionnez la politique que vous avez créée.
7. Choisissez Suivant.
8. Dans le champ Role name (Nom de rôle), saisissez un nom pour votre rôle. Pour cet exemple, tapez `AmazonECSTaskS3BucketRole` comme nom du rôle.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit : Remplacez l'identifiant de région et spécifiez AWS numéro de compte que vous utilisez pour lancer des tâches.

   JSON

```
{
   "Version":"2012-10-17",
   "Statement":[
      {
         "Effect":"Allow",
         "Principal":{
            "Service":[
               "ecs-tasks.amazonaws.com"
            ]
         },
         "Action":"sts:AssumeRole",
         "Condition":{
            "ArnLike":{
            "aws:SourceArn":"arn:aws:ecs:us-west-2:111122223333:*"
            },
            "StringEquals":{
               "aws:SourceAccount":"111122223333"
            }
         }
      }
   ]
}
```

2. Créez un rôle IAM nommé `ecsTaskRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsTaskRole \
      --assume-role-policy-document file://ecs-tasks-trust-policy.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante. Remplacer *taskRolePolicy* avec le nom de la politique que vous avez créée.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`taskRolePolicy`].Arn'
```

4. Attachez la politique IAM que vous avez créée au rôle `ecsTaskRole`. Remplacez `policy-arn` par l'ARN de la politique que vous avez créée.

```
aws iam attach-role-policy \
      --role-name ecsTaskRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/taskRolePolicy
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Utiliser ECS Exec	Autorisations ECS Exec
Utiliser une image provenant d'un référentiel Amazon ECR privé	Autorisations Amazon ECR
Utiliser des instances EC2 (Windows et Linux)	Configuration supplémentaire des instances Amazon EC2
Utiliser des instances externes	Configuration supplémentaire d'instance externe
Utiliser des instances Windows EC2	Configuration supplémentaire d'instance Windows Amazon EC2

Autorisations Amazon ECR

Les autorisations suivantes sont requises lorsque le code de votre application doit interagir directement avec les référentiels Amazon ECR. Notez que pour une implémentation de base où vous n'avez besoin que d'extraire des images d'Amazon ECR, ces autorisations ne sont pas requises au niveau du rôle IAM de tâche. Au lieu de cela, le rôle d'exécution de tâche Amazon ECS doit avoir ces autorisations. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Si le code de votre application exécuté dans le conteneur doit interagir directement avec les API Amazon ECR, vous devez ajouter les autorisations suivantes à un rôle IAM de tâche et inclure le rôle IAM de tâche dans votre définition de tâche. Pour obtenir des informations, consultez la section Ajout et suppression de politiques IAM dans le Guide de l'utilisateur IAM.

Utilisez la politique suivante pour votre rôle IAM de tâche afin d'ajouter les autorisations Amazon ECR requises pour les applications de conteneur qui doivent interagir directement avec Amazon ECR :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Autorisations ECS Exec

La fonctionnalité ECS Exec nécessite un rôle IAM de tâche pour octroyer aux conteneurs les autorisations nécessaires à la communication entre l'agent SSM géré (agent `execute-command`) et le service SSM. Vous devez ajouter les autorisations suivantes à un rôle IAM de tâche et inclure le

rôle IAM de tâche dans votre définition de tâche. Pour obtenir des informations, consultez la section Ajout et suppression de politiques IAM dans le Guide de l'utilisateur IAM.

Utilisez la stratégie suivante pour votre rôle IAM de tâche afin d'ajouter les autorisations SSM requises.

JSON

```
{
   "Version":"2012-10-17",
   "Statement": [
       {
       "Effect": "Allow",
       "Action": [
            "ssmmessages:CreateControlChannel",
            "ssmmessages:CreateDataChannel",
            "ssmmessages:OpenControlChannel",
            "ssmmessages:OpenDataChannel"
       ],
      "Resource": "*"
      }
   ]
}
```

Configuration supplémentaire des instances Amazon EC2

Nous vous recommandons de limiter les autorisations dans votre rôle d'instance de conteneur à la liste minimale des autorisations utilisées dans la politique IAM gérée `AmazonEC2ContainerServiceforEC2Role`.

Pour utiliser les rôles IAM de tâche, vos instances Amazon EC2 doivent au moins disposer de la version `1.11.0` de l'agent de conteneur ; nous vous conseillons cependant d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez un Amazon ECS-optimized AMI, votre instance a besoin d'au moins `1.11.0-1` de la `ecs-init` colis. Si vos instances utilisent la dernière version d'Amazon ECS-optimized AMI, ils contiennent alors les versions requises de l'agent de conteneur et `ecs-init`. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

Si vous n'utilisez pas Amazon ECS-optimized AMI pour vos instances de conteneur, ajoutez le `--net=host` option pour votre docker run commande qui démarre l'agent et les variables de configuration d'agent suivantes pour la configuration souhaitée (pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS) :

`ECS_ENABLE_TASK_IAM_ROLE=true`

Utilise les rôles IAM pour les tâches des conteneurs associés aux modes réseau `bridge` et `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Utilise les rôles IAM pour les tâches des conteneurs associés au mode réseau `host`. Cette variable est uniquement prise en charge sur les versions d'agent 1.12.0 et ultérieures.

Pour obtenir un exemple de commande d'exécution, consultez Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-Optimized AMI). Vous devrez également définir les commandes réseau suivantes sur votre instance de conteneur afin que les conteneurs de vos tâches puissent récupérer leur AWS informations d'identification :

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Pour garantir le maintien de ces règles iptables après un redémarrage, vous devez les enregistrer sur votre instance de conteneur. Vous pouvez utiliser les commandes iptables-save et iptables-restore pour enregistrer vos règles iptables et les restaurer au démarrage. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Pour empêcher les conteneurs exécutés par des tâches utilisant le mode réseau `awsvpc` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, définissez la variable de configuration d'agent `ECS_AWSVPC_BLOCK_IMDS` sur `true` dans le fichier de configuration de l'agent et redémarrez l'agent. Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS.

Pour empêcher les conteneurs exécutés par des tâches utilisant le mode réseau `bridge` d'accéder aux informations d'identification fournies par le profil d'instance Amazon EC2, tout en accordant les autorisations fournies par le rôle de la tâche, exécutez la commande iptables suivante sur vos

instances Amazon EC2. Cette commande n'a pas d'incidence sur les conteneurs dans les tâches utilisant le mode réseau `host` ou `awsvpc`. Pour plus d'informations, consultez Mode réseau.

- ```
  sudo yum install -y iptables-services; sudo iptables --insert DOCKER-USER 1 --in-interface docker+ --destination 169.254.169.254/32 --jump DROP
  ```

  Pour garantir le maintien de la règle iptables après un redémarrage, vous devez l'enregistrer sur votre instance Amazon EC2. Lorsque vous utilisez Amazon ECS-optimized AMI, vous pouvez utiliser la commande suivante. Pour les autres systèmes d'exploitation, consultez la documentation correspondante.

  ```
  sudo iptables-save | sudo tee /etc/sysconfig/iptables && sudo systemctl enable --now iptables
  ```

Configuration supplémentaire d'instance externe

Pour utiliser les rôles IAM de tâche, vos instances externes doivent au moins disposer de la version `1.11.0` de l'agent de conteneur ; nous vous conseillons cependant d'utiliser la dernière version de l'agent de conteneur. Pour plus d'informations sur la vérification de la version de votre agent et la mise à jour à la dernière version, consultez Mise à jour de l'agent de conteneur Amazon ECS. Si vous utilisez un Amazon ECS-optimized AMI, votre instance a besoin d'au moins `1.11.0-1` de la `ecs-init` colis. Si vos instances utilisent la dernière version d'Amazon ECS-optimized AMI, ils contiennent alors les versions requises de l'agent de conteneur et `ecs-init`. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.

Si vous n'utilisez pas Amazon ECS-optimized AMI pour vos instances de conteneur, ajoutez le `--net=host` option pour votre docker run commande qui démarre l'agent et les variables de configuration d'agent suivantes pour la configuration souhaitée (pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS) :

`ECS_ENABLE_TASK_IAM_ROLE=true`

Utilise les rôles IAM pour les tâches des conteneurs associés aux modes réseau `bridge` et `default`.

`ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST=true`

Utilise les rôles IAM pour les tâches des conteneurs associés au mode réseau `host`. Cette variable est uniquement prise en charge sur les versions d'agent 1.12.0 et ultérieures.
  ```

Pour obtenir un exemple de commande d'exécution, consultez Mise à jour manuelle de l'agent de conteneur Amazon ECS (pour les non-Amazon) ECS-Optimized AMI). Vous devrez également définir les commandes réseau suivantes sur votre instance de conteneur afin que les conteneurs de vos tâches puissent récupérer leur AWS informations d'identification :

```
sudo sysctl -w net.ipv4.conf.all.route_localnet=1
sudo iptables -t nat -A PREROUTING -p tcp -d 169.254.170.2 --dport 80 -j DNAT --to-
destination 127.0.0.1:51679
sudo iptables -t nat -A OUTPUT -d 169.254.170.2 -p tcp -m tcp --dport 80 -j REDIRECT --
to-ports 51679
```

Pour garantir le maintien de ces règles iptables après un redémarrage, vous devez les enregistrer sur votre instance de conteneur. Vous pouvez utiliser les commandes iptables-save et iptables-restore pour enregistrer vos règles iptables et les restaurer au démarrage. Pour plus d'informations, consultez la documentation de votre système d'exploitation.

Configuration supplémentaire d’instance Windows Amazon EC2

> **Important**
>
> Cela ne s’applique qu’aux conteneurs Windows sur EC2 qui utilisent des rôles de tâche.

Le rôle de tâche associé aux fonctionnalités Windows nécessite une configuration supplémentaire sur EC2.

- Lorsque vous lancez vos instances de conteneur, vous devez définir l'option `EnableTaskIAMRole` dans le script de données utilisateur des instances de conteneur. `EnableTaskIAMRole` active la fonction Rôles IAM pour les tâches. Par exemple :

  ```
  <powershell>
  Import-Module ECSTools
  Initialize-ECSAgent -Cluster 'windows' -EnableTaskIAMRole
  </powershell>
  ```

- Vous devez amorcer votre conteneur avec les commandes de mise en réseau fournies dans Script d’amorçage de conteneur Amazon ECS.
- Vous devez créer un rôle et une stratégie IAM pour vos tâches. Pour plus d’informations, consultez Création du rôle IAM de tâche.

- Les rôles IAM pour le fournisseur d'informations d'identification de tâche utilisent le port 80 sur l'instance de conteneur. Par conséquent, si vous configurez des rôles IAM pour les tâches sur votre instance de conteneur, vos conteneurs ne peuvent pas utiliser le port 80 pour le port hôte des mappages de ports. Pour exposer vos conteneurs sur le port 80, nous vous recommandons de configurer un service pour ceux qui utilisent l'équilibrage de charge. Vous pouvez utiliser le port 80 sur l'équilibreur de charge. Ce faisant, le trafic peut être acheminé vers un autre port hôte sur vos instances de conteneur. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.
- Si votre instance Windows est redémarrée, vous devez supprimer l'interface proxy et initialiser à nouveau l'agent conteneur Amazon ECS pour rétablir la sauvegarde du proxy d'informations d'identification.

Script d'amorçage de conteneur Amazon ECS

Avant que les conteneurs puissent accéder au proxy d'informations d'identification sur l'instance de conteneur afin d'obtenir des informations d'identification, le conteneur doit être amorcé avec les commandes de mise en réseau requises. L'exemple de script de code suivant doit être exécuté sur vos conteneurs lorsqu'ils démarrent.

> **Note**
>
> Vous n'avez pas besoin d'exécuter ce script lorsque vous utilisez le mode réseau `awsvpc` sur Windows.

Si vous exécutez des conteneurs Windows qui incluent Powershell, utilisez le script suivant :

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.
```

```
$gateway = (Get-NetRoute | Where { $_.DestinationPrefix -eq '0.0.0.0/0' } | Sort-Object
 RouteMetric | Select NextHop).NextHop
$ifIndex = (Get-NetAdapter -InterfaceDescription "Hyper-V Virtual Ethernet*" | Sort-
Object | Select ifIndex).ifIndex
New-NetRoute -DestinationPrefix 169.254.170.2/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # credentials API
New-NetRoute -DestinationPrefix 169.254.169.254/32 -InterfaceIndex $ifIndex -NextHop
 $gateway -PolicyStore ActiveStore # metadata API
```

Si vous exécutez des conteneurs Windows qui n'ont que le shell Command, utilisez le script suivant :

```
# Copyright Amazon.com Inc. or its affiliates. All Rights Reserved.
#
# Licensed under the Apache License, Version 2.0 (the "License"). You may
# not use this file except in compliance with the License. A copy of the
# License is located at
#
# http://aws.amazon.com/apache2.0/
#
# or in the "license" file accompanying this file. This file is distributed
# on an "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either
# express or implied. See the License for the specific language governing
# permissions and limitations under the License.

for /f "tokens=1" %i  in ('netsh interface ipv4 show interfaces ^| findstr /x /r
 ".*vEthernet.*"') do set interface=%i
for /f "tokens=3" %i  in ('netsh interface ipv4 show addresses %interface% ^| findstr /
x /r ".*Default.Gateway.*"') do set gateway=%i
netsh interface ipv4 add route prefix=169.254.170.2/32 interface="%interface%"
 nexthop="%gateway%" store=active # credentials API
netsh interface ipv4 add route prefix=169.254.169.254/32 interface="%interface%"
 nexthop="%gateway%" store=active # metadata API
```

Rôle IAM d'instance de conteneur Amazon ECS

Les instances de conteneur Amazon ECS, y compris Amazon EC2 et les instances externes, exécutent l'agent de conteneur Amazon ECS et utilisent un rôle IAM pour indiquer au service que l'agent vous appartient. Avant de pouvoir lancer des instances de conteneur et de les enregistrer dans un cluster, vous devez créer le rôle IAM qu'elles utiliseront. Le rôle est créé dans le compte que vous utilisez pour vous connecter à la console ou exécuter le AWS CLI commandes.

> **Important**
>
> Si vous enregistrez des instances externes dans votre cluster, le rôle IAM que vous utilisez nécessite également des autorisations Systems Manager. Pour plus d'informations, consultez Rôle IAM Amazon ECS Anywhere.

Amazon ECS fournit le fichier de stratégie IAM gérée `AmazonEC2ContainerServiceforEC2Role` qui contient les autorisations nécessaires à l'utilisation de l'ensemble complet de fonctions Amazon ECS. Cette stratégie gérée peut être attachée à un rôle IAM et associée à vos instances de conteneur. Sinon, vous pouvez utiliser la stratégie gérée comme guide lorsque vous créez une stratégie personnalisée à utiliser. Le rôle d'instance de conteneur fournit les autorisations nécessaires pour que l'agent de conteneur Amazon ECS et le démon Docker puissent appeler AWS Des API en votre nom. Pour de plus amples informations sur la stratégie gérée, veuillez consulter AmazonEC2ContainerServiceforEC2Role.

Création du rôle d'instance de conteneur

> **Important**
>
> Si vous enregistrez des instances externes dans votre cluster, veuillez consulter Rôle IAM Amazon ECS Anywhere.

Vous pouvez créer et associer manuellement le rôle à la politique IAM gérée pour les instances de conteneur afin de permettre à Amazon ECS d'ajouter des autorisations pour les futures fonctionnalités et améliorations au fur et à mesure qu'elles sont introduites. Utilisez la procédure suivante pour attacher la politique IAM gérée si nécessaire.

Console de gestion AWS

Pour créer un rôle de service Elastic Container Service (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Rôles, puis Créer un rôle.
3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.

4. Pour Service ou cas d'utilisation, choisissez Elastic Container Service, puis choisissez le cas d'utilisation Rôle EC2 pour Elastic Container Service.
5. Choisissez Suivant.
6. Dans le Politiques relatives aux autorisations section, vérifiez que AmazonEC2ContainerServiceforEC2Role la politique est sélectionnée.

   > **Important**
   >
   > Le AmazonEC2ContainerServiceforEC2Role la politique gérée doit être attachée au rôle IAM de l'instance de conteneur, sinon vous recevrez une erreur en utilisant le Console de gestion AWS pour créer des clusters.

7. Choisissez Suivant.
8. Pour Nom du rôle, entrez ecsInstanceRole
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tous les *user input* par vos propres valeurs.

1. Créez un fichier nommé `instance-role-trust-policy.json` avec le contenu suivant.

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Principal": { "Service": "ec2.amazonaws.com"},
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez la commande suivante pour créer le rôle IAM de l'instance à l'aide du document de stratégie d'approbation.

```
aws iam create-role \
    --role-name ecsInstanceRole \
```

```
    --assume-role-policy-document file://instance-role-trust-policy.json
```

3. Créez un profil d'instance nommé `ecsInstanceRole-profile` à l'aide de la commande create-instance-profile.

```
aws iam create-instance-profile --instance-profile-name ecsInstanceRole-profile
```

Exemple de réponse

```
{
    "InstanceProfile": {
        "InstanceProfileId": "AIPAJTLBPJLEGREXAMPLE",
        "Roles": [],
        "CreateDate": "2022-04-12T23:53:34.093Z",
        "InstanceProfileName": "ecsInstanceRole-profile",
        "Path": "/",
        "Arn": "arn:aws:iam::123456789012:instance-profile/ecsInstanceRole-
profile"
    }
}
```

4. Ajoutez le rôle *ecsInstanceRole* au profil d'instance *ecsInstanceRole-profile*.

```
aws iam add-role-to-instance-profile \
    --instance-profile-name ecsInstanceRole-profile \
    --role-name ecsInstanceRole
```

5. Attachez la politique gérée par `AmazonEC2ContainerServiceForEC2Role` au rôle à l'aide de la commande suivante.

```
aws iam attach-role-policy \
    --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceforEC2Role \
    --role-name ecsInstanceRole
```

Après avoir créé le rôle, ajoutez-y des autorisations supplémentaires pour les fonctionnalités suivantes.

Fonctionnalité	Autorisations supplémentaires
Amazon ECR contient l'image du conteneur	Autorisations Amazon ECR
Avoir CloudWatch Les journaux surveillent les instances de conteneurs	Autorisations pour la surveillance des instances de conteneur
Héberger les fichiers de configuration dans un compartiment Amazon S3	Accès en lecture seule à Amazon S3

Autorisations Amazon ECR

Le rôle d'instance de conteneur Amazon ECS que vous utilisez avec vos instances de conteneur doit disposer des autorisations de politique IAM suivantes pour Amazon ECR.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecr:BatchCheckLayerAvailability",
                "ecr:BatchGetImage",
                "ecr:GetDownloadUrlForLayer",
                "ecr:GetAuthorizationToken"
            ],
            "Resource": "*"
        }
    ]
}
```

Si vous utilisez la stratégie gérée `AmazonEC2ContainerServiceforEC2Role` pour vos instances de conteneurs, votre rôle possède les autorisations appropriées.

Autorisations requises pour définir le paramètre du compte awsvpcTrunking

Amazon ECS prend en charge le lancement d'instances de conteneur avec une augmentation de la densité ENI et utilisant des types d'instances Amazon EC2 pris en charge. Lorsque vous utilisez cette fonctionnalité, nous vous recommandons de créer deux rôles d'instance de conteneur. Activez le paramètre de compte `awsvpcTrunking` pour un rôle et utilisez ce rôle pour les tâches qui nécessitent la jonction ENI. Pour en savoir plus sur le paramètre de compte `awsvpcTrunking`, consultez la section Accès aux fonctionnalités d'Amazon ECS avec les paramètres du compte.

Le rôle d'instance de conteneur que vous utilisez avec vos instances de conteneur doit disposer des autorisations de politique IAM suivantes pour définir les paramètres de compte

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:ListAccountSettings",
                "ecs:ListAttributes",
                "ecs:PutAccountSetting"
            ],
            "Resource": "*"
        }
    ]
}
```

Pour utiliser le rôle d'instance de conteneur, ajoutez ce qui suit aux données utilisateur de l'instance :

```
#!/bin/bash
aws ecs put-account-setting --name awsvpcTrunking --value enabled --region region
ECS_CLUSTER=MyCluster>> /etc/ecs/ecs.config
EOF
```

Pour plus d'informations sur l'ajout de données utilisateur à vos instances EC2, consultez la section Exécuter des commandes sur votre instance Linux au démarrage dans le Guide de l'utilisateur Amazon EC2.

Accès en lecture seule à Amazon S3

Stocker les informations de configuration dans un compartiment privé d'Amazon S3 et accorder un accès en lecture seule au rôle IAM de votre instance de conteneur est un moyen pratique et sécurisé d'autoriser la configuration d'une instance de conteneur au moment du lancement. Vous pouvez enregistrer une copie de votre `ecs.config` fichier dans un compartiment privé, utilisez les données utilisateur d'Amazon EC2 pour installer AWS CLI puis copiez vos informations de configuration dans `/etc/ecs/ecs.config` lors du lancement de l'instance.

Pour plus d'informations sur la création d'un fichier `ecs.config`, son stockage dans Amazon S3 et le lancement d'instances avec cette configuration, veuillez consulter Stockage de la configuration d'instance de conteneur Amazon ECS dans Amazon S3.

Vous pouvez utiliser ce qui suit AWS CLI commande pour autoriser l'accès en lecture seule à Amazon S3 pour votre rôle d'instance de conteneur. Remplacer *`ecsInstanceRole`* avec le nom du rôle que vous avez créé.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
      --policy-arn arn:aws:iam::aws:policy/AmazonS3ReadOnlyAccess
```

Vous pouvez également utiliser la console IAM pour ajouter l'accès en lecture seule à Amazon S3 (`AmazonS3ReadOnlyAccess`) à votre rôle. Pour plus d'informations, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .

Autorisations pour la surveillance des instances de conteneur

Avant que vos instances de conteneur puissent envoyer des données de journal à CloudWatch Journaux, vous devez créer une politique IAM pour permettre à l'agent Amazon ECS d'écrire les journaux des applications du client dans CloudWatch (normalement géré par le `awslogs` chauffeur). Après avoir créé la politique, associez-la à `ecsInstanceRole`.

Console de gestion AWS

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
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 Mise en route.

3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

6. 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 plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez 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.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la politique, associez-la au rôle d'instance de conteneur. Pour plus d'informations sur la manière d'associer une politique à un rôle, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .

AWS CLI

1. Créez un fichier nommé `instance-cw-logs.json` avec le contenu suivant.

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "logs:CreateLogGroup",
                "logs:CreateLogStream",
                "logs:PutLogEvents",
                "logs:DescribeLogStreams"
            ],
            "Resource": ["arn:aws:logs:*:*:*"]
        }
    ]
}
```

2. Utilisez la commande suivante pour créer la politique IAM à l'aide du fichier de politique JSON.

```
aws iam create-policy \
      --policy-name cwlogspolicy \
      --policy-document file://instance-cw-logs.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante. Remplacer *cwlogspolicy* avec le nom de la politique que vous avez créée.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cwlogspolicy`].Arn'
```

4. Utilisez la commande suivante pour associer la politique au rôle IAM de l'instance de conteneur à l'aide de l'ARN de la politique.

```
aws iam attach-role-policy \
      --role-name ecsInstanceRole \
```

```
    --policy-arn arn:aws:iam:111122223333:aws:policy/cwlogspolicy
```

Rôle IAM Amazon ECS Anywhere

Lorsque vous enregistrez un serveur ou une machine virtuelle (VM) sur site dans votre cluster, le serveur ou la machine virtuelle nécessite un rôle IAM pour communiquer avec AWS API. Vous n'avez besoin de créer ce rôle IAM qu'une seule fois pour chaque AWS compte. Toutefois, ce rôle IAM doit être associé à chaque serveur ou machine virtuelle que vous enregistrez sur un cluster. Ce rôle est le `ECSAnywhereRole`. Vous pouvez créer ce rôle manuellement. De même, Amazon ECS peut créer le rôle en votre nom lorsque vous enregistrez une instance externe dans la Console de gestion AWS. Vous pouvez utiliser la recherche de la console IAM pour rechercher `ecsAnywhereRole` et voir si votre compte dispose déjà de ce rôle. Pour plus d'informations, consultez la section Recherche dans la console IAM dans le Guide de l'utilisateur IAM.

AWS fournit deux politiques IAM gérées qui peuvent être utilisées lors de la création du rôle IAM ECS Anywhere, le `AmazonSSMManagedInstanceCore` et `AmazonEC2ContainerServiceforEC2Role` politiques. La stratégie `AmazonEC2ContainerServiceforEC2Role` inclut des autorisations qui fournissent probablement plus d'accès que vous n'avez besoin. Par conséquent, en fonction de votre cas d'utilisation spécifique, nous vous recommandons de créer une stratégie personnalisée en ajoutant uniquement les autorisations de cette stratégie dont vous avez besoin. Pour de plus amples informations, consultez la section Rôle IAM d'instance de conteneur Amazon ECS.

Le rôle IAM d'exécution de tâche qui accorde à l'agent de conteneur Amazon ECS l'autorisation d'effectuer des appels d'API AWS en votre nom. Lorsqu'un rôle IAM d'exécution de tâche est utilisé, il doit être spécifié dans votre définition de tâche. Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.

Le rôle d'exécution de tâche est requis si l'une des conditions suivantes s'applique :

- Vous envoyez des journaux de conteneurs à CloudWatch Journaux à l'aide du `awslogs` pilote de journaux.
- Votre définition de tâche spécifie une image de conteneur hébergée dans un référentiel privé Amazon ECR. Cependant, si le rôle `ECSAnywhereRole` associé à votre instance externe inclut également les autorisations nécessaires pour extraire des images d'Amazon ECR, alors votre rôle d'exécution de tâche n'a pas besoin de les inclure.

Création du rôle Amazon ECS Anywhere

Remplacez tout *user input* avec vos propres informations.

1. Créez un fichier local nommé `ssm-trust-policy.json` avec la stratégie d'approbation suivante :

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": {
       "Effect": "Allow",
       "Principal": {"Service": [
         "ssm.amazonaws.com"
       ]},
       "Action": "sts:AssumeRole"
     }
   }
   ```

2. Créez le rôle et associez-lui la stratégie d'approbation à l'aide de la commande AWS CLI suivante.

   ```
   aws iam create-role --role-name ecsAnywhereRole --assume-role-policy-document
    file://ssm-trust-policy.json
   ```

3. Joignez le AWS politiques gérées à l'aide de la commande suivante.

   ```
   aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
    arn:aws:iam::aws:policy/AmazonSSMManagedInstanceCore
   aws iam attach-role-policy --role-name ecsAnywhereRole --policy-arn
    arn:aws:iam::aws:policy/service-role/AmazonEC2ContainerServiceforEC2Role
   ```

Vous pouvez également utiliser le flux de travail de stratégie d'approbation personnalisé IAM pour créer le rôle. Pour plus d'informations, reportez-vous à la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le Guide de l'utilisateur IAM.

Rôle IAM d’infrastructure Amazon ECS

Un rôle IAM d’infrastructure Amazon ECS permet à Amazon ECS de gérer les ressources d’infrastructure de vos clusters en votre nom. Il est utilisé dans les cas suivants :

- Vous souhaitez associer des volumes Amazon EBS à vos tâches Amazon ECS dont le type de lancement est Fargate ou EC2. Le rôle d’infrastructure permet à Amazon ECS de gérer les volumes Amazon EBS pour vos tâches.

  Vous pouvez utiliser la politique gérée `AmazonECSInfrastructureRolePolicyForVolumes`
- Vous souhaitez utiliser le protocole TLS (Transport Layer Security) pour chiffrer le trafic entre vos services Amazon ECS Service Connect.

  Vous pouvez utiliser la politique gérée `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`
- Vous souhaitez créer des groupes cibles Amazon VPC Lattice.

  Vous pouvez utiliser la politique gérée `AmazonECSInfrastructureRolePolicyForVpcLattice`
- Vous souhaitez utiliser des instances gérées Amazon ECS dans vos clusters Amazon ECS. Le rôle d’infrastructure permet à Amazon ECS de gérer le cycle de vie des instances gérées.

  Vous pouvez utiliser la politique gérée `AmazonECSInfrastructureRolePolicyForManagedInstances`
- Vous souhaitez utiliser le mode Express. Le rôle d'infrastructure permet à Amazon ECS de provisionner et de gérer les composants d'infrastructure requis pour les services en mode Express, notamment l'équilibrage de charge, les groupes de sécurité, les certificats SSL et les configurations de dimensionnement automatique.

  Vous pouvez utiliser la politique gérée `AmazonECSInfrastructureRoleforExpressGatewayServices`

Lorsqu’Amazon ECS assume ce rôle pour prendre des mesures en votre nom, les événements seront visibles dans AWS CloudTrail. Si Amazon ECS utilise ce rôle pour gérer les volumes Amazon EBS associés à vos tâches, CloudTrail bûche `roleSessionName` sera `ECSTaskVolumesForEBS`. Si le rôle est utilisé pour chiffrer le trafic entre vos services Service Connect, le CloudTrail bûche `roleSessionName` sera `ECSServiceConnectForTLS`. Si le rôle est utilisé pour créer des groupes cibles pour VPC Lattice, CloudTrail bûche `roleSessionName` sera

`ECSNetworkingWithVPCLattice`. Si le rôle est utilisé pour gérer les instances gérées Amazon ECS, CloudTrail bûche `roleSessionName` sera `ECSManagedInstancesForCompute`. Vous pouvez utiliser ce nom pour rechercher des événements dans CloudTrail console en filtrant pour Nom d'utilisateur.

Amazon ECS fournit des politiques gérées qui contiennent les autorisations requises pour l'attachement de volumes, le protocole TLS, VPC Lattice et les instances gérées. Pour plus d'informations, voir AmazonECSInfrastructureRolePolicyForVolumes, AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity, AmazonECSInfrastructureRolePolicyForVpcLattice, AmazonECSInfrastructureRolePolicyForManagedInstances, et AmazonECSInfrastructureRoleforExpressGatewayServices dans le AWS Guide de référence des politiques gérées.

Création du rôle d'infrastructure Amazon ECS

Remplacez tout *`user input`* avec vos propres informations.

1. Créez un fichier nommé `ecs-infrastructure-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez ce qui suit AWS CLI commande pour créer un rôle nommé `ecsInfrastructureRole` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsInfrastructureRole \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. En fonction de votre cas d'utilisation, associez la politique gérée au `ecsInfrastructureRole` rôle.

   - Pour associer des volumes Amazon EBS à vos tâches Amazon ECS de type de lancement Fargate ou EC2, joignez le `AmazonECSInfrastructureRolePolicyForVolumes` politique gérée.
   - Pour utiliser le protocole TLS (Transport Layer Security) afin de chiffrer le trafic entre vos services Amazon ECS Service Connect, joignez le `AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity` politique gérée.
   - Pour créer des groupes cibles Amazon VPC Lattice, joignez le `AmazonECSInfrastructureRolePolicyForVpcLattice` politique gérée.
   - Si vous souhaitez utiliser les instances gérées Amazon ECS dans vos clusters Amazon ECS, joignez le `AmazonECSInfrastructureRolePolicyForManagedInstances` politique gérée.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForVolumes
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity
```

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRole \
      --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRolePolicyForManagedInstances
```

Vous pouvez également utiliser le flux de travail Stratégie d'approbation personnalisée de la console IAM pour créer le rôle. Pour plus d'informations, reportez-vous à la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le Guide de l'utilisateur IAM.

Important

Si le rôle d'infrastructure est utilisé par Amazon ECS pour gérer les volumes Amazon EBS associés à vos tâches, assurez-vous de ce qui suit avant d'arrêter les tâches qui utilisent des volumes Amazon EBS.

- Le rôle n'est pas supprimé.
- La stratégie d'approbation associée au rôle n'est pas modifiée pour supprimer l'accès Amazon ECS (`ecs.amazonaws.com`).
- La politique gérée `AmazonECSInfrastructureRolePolicyForVolumes` n'est pas supprimée. Si vous devez modifier les autorisations du rôle, conservez au moins `ec2:DetachVolume`, `ec2:DeleteVolume` et `ec2:DescribeVolumes` pour la suppression du volume.

Si vous supprimez ou modifiez le rôle avant d'arrêter les tâches associées à des volumes Amazon EBS, les tâches resteront bloquées en `DEPROVISIONING` et les volumes Amazon EBS associés ne seront pas supprimés. Amazon ECS réessaiera automatiquement à intervalles réguliers d'arrêter la tâche et de supprimer le volume jusqu'à ce que les autorisations nécessaires soient rétablies. Vous pouvez consulter l'état des pièces jointes au volume d'une tâche et la raison du statut associé à l'aide du DescribeTasks API.

Une fois le fichier créé, vous devez autoriser votre utilisateur à transmettre le rôle à Amazon ECS.

Autorisation de transmettre le rôle d'infrastructure à Amazon ECS

Pour utiliser un rôle IAM d'infrastructure ECS, vous devez accorder à votre utilisateur l'autorisation de transmettre le rôle à Amazon ECS. Attachez les autorisations `iam:PassRole` suivantes à votre utilisateur. Remplacer *`ecsInfrastructureRole`* avec le nom du rôle d'infrastructure que vous avez créé.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [

        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource": ["arn:aws:iam::*:role/ecsInfrastructureRole"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
    ]
}
```

Pour plus d'informations sur `iam:Passrole` et pour mettre à jour les autorisations de votre utilisateur, consultez Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS service et Modifier les autorisations d'un utilisateur IAM dans le Gestion des identités et des accès AWS Guide de l'utilisateur.

Profil d’instance des instances gérées Amazon ECS

Un profil d’instance est un conteneur IAM qui contient exactement un rôle IAM et permet aux instances gérées Amazon ECS d’assumer ce rôle en toute sécurité. Le profil d’instance contient un rôle d’instance que l’agent ECS assume pour enregistrer les instances auprès des clusters et communiquer avec le service ECS.

Important

Si vous utilisez des instances gérées Amazon ECS avec `AmazonECSInfrastructureRolePolicyForManagedInstances` politique gérée, le nom du rôle d'instance doit commencer par `ecsInstanceRole`. Les champs d'application de la politique `iam:PassRole` pour `arn:aws:iam::*:role/ecsInstanceRole*`, donc un nom différent provoque une erreur d'autorisation lors du lancement de la tâche. Ceci est courant avec CloudFormation lorsque vous omettez `RoleName`, parce que CloudFormation génère automatiquement des noms tels que `MyStack-InstanceRole-ABC123`.

Si vous utilisez plutôt une politique de rôle d'infrastructure personnalisée, le rôle d'instance peut avoir n'importe quel nom tant que votre stratégie inclut un `iam:PassRole` subvention ciblant l'ARN du rôle d'instance.

Création d'un rôle avec la stratégie d'approbation

Remplacez tout *`user input`* avec vos propres informations.

1. Créez un fichier nommé `ecsInstanceRole-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

   ```
   {
     "Version":"2012-10-17",
     "Statement": [
       {
         "Effect": "Allow",
         "Principal": { "Service": "ec2.amazonaws.com"},
         "Action": "sts:AssumeRole"
       }
     ]
   }
   ```

2. Utilisez ce qui suit AWS CLI commande pour créer un rôle nommé `ecsInstanceRole` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

   ```
   aws iam create-role \
         --role-name ecsInstanceRole \
         --assume-role-policy-document file://ecsInstanceRole-trust-policy.json
   ```

3. Joignez le AWS géré `AmazonECSInstanceRolePolicyForManagedInstances` politique à l'égard de `ecsInstanceRole` rôle.

   ```
   aws iam attach-role-policy \
         --role-name ecsInstanceRole \
         --policy-arn arn:aws:iam::aws:policy/
   AmazonECSInstanceRolePolicyForManagedInstances
   ```

> **Note**
>
> Si vous choisissez d'appliquer les autorisations de moindre privilège et de spécifier vos propres autorisations à la place, vous pouvez ajouter les autorisations suivantes pour vous aider à résoudre les problèmes liés aux tâches avec les instances gérées Amazon ECS :
>
> - `ecs:StartTelemetrySession`
> - `ecs:PutSystemLogEvents`

Vous pouvez également utiliser le flux de travail Stratégie d'approbation personnalisée de la console IAM pour créer le rôle. Pour plus d'informations, reportez-vous à la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le Guide de l'utilisateur IAM.

Une fois le fichier créé, vous devez autoriser votre utilisateur à transmettre le rôle à Amazon ECS.

Créez le profil d'instance à l'aide du AWS CLI

Après avoir créé le rôle, créez le profil d'instance à l'aide de l' AWS CLI :

```
aws iam create-instance-profile --instance-profile-name ecsInstanceRole
```

Ajoutez le rôle au profil d'instance :

```
aws iam add-role-to-instance-profile \
   --instance-profile-name ecsInstanceRole \
   --role-name ecsInstanceRole
```

Vérifiez que le profil a été créé avec succès :

```
aws iam get-instance-profile --instance-profile-name ecsInstanceRole
```

Créez le profil d'instance à l'aide de CloudFormation

Vous pouvez utiliser AWS CloudFormation pour créer le rôle et le profil d'instance. Choisissez l'une des options suivantes selon que vous utilisez ou non AWS-une politique d'infrastructure gérée ou une politique personnalisée.

Option 1 : utiliser l'ecsInstanceRole convention de dénomination (recommandée)

Lorsque vous utilisez AWS-politique d'infrastructure gérée, vous devez définir explicitement `RoleName` à une valeur commençant par `ecsInstanceRole`. Si vous omettez `RoleName`, CloudFormation génère automatiquement un nom qui ne correspond pas à celui de la politique gérée `iam:PassRole` condition, et les tâches ne peuvent pas être lancées.

```
Resources:
  EcsInstanceRole:
    Type: AWS::IAM::Role
    Properties:
      RoleName: ecsInstanceRole
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
        Statement:
          - Effect: Allow
            Principal:
              Service: ec2.amazonaws.com
            Action: sts:AssumeRole
      ManagedPolicyArns:
        - arn:aws:iam::aws:policy/AmazonECSInstanceRolePolicyForManagedInstances

  EcsInstanceProfile:
    Type: AWS::IAM::InstanceProfile
    Properties:
      InstanceProfileName: ecsInstanceRole
      Roles:
        - !Ref EcsInstanceRole
```

Option 2 : utiliser un nom de rôle personnalisé

Si vous préférez louer CloudFormation générez le nom du rôle, ou vous utilisez un nom personnalisé qui ne commence pas par `ecsInstanceRole`, vous devez ajouter une politique en ligne à votre rôle d'infrastructure qui accorde `iam:PassRole` pour le rôle d'instance.

```
Resources:
  EcsInstanceRole:
    Type: AWS::IAM::Role
    Properties:
      # No RoleName — CFN auto-generates
      AssumeRolePolicyDocument:
        Version: "2012-10-17"
```

```
      Statement:
        - Effect: Allow
          Principal:
            Service: ec2.amazonaws.com
          Action: sts:AssumeRole
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/AmazonECSInstanceRolePolicyForManagedInstances

EcsInstanceProfile:
  Type: AWS::IAM::InstanceProfile
  Properties:
    Roles:
      - !Ref EcsInstanceRole

EcsInfrastructureRole:
  Type: AWS::IAM::Role
  Properties:
    AssumeRolePolicyDocument:
      Version: "2012-10-17"
      Statement:
        - Effect: Allow
          Principal:
            Service: ecs.amazonaws.com
          Action: sts:AssumeRole
    ManagedPolicyArns:
      - arn:aws:iam::aws:policy/AmazonECSInfrastructureRolePolicyForManagedInstances
    Policies:
      - PolicyName: PassInstanceRoleToEC2
        PolicyDocument:
          Version: "2012-10-17"
          Statement:
            - Effect: Allow
              Action: iam:PassRole
              Resource: !GetAtt EcsInstanceRole.Arn
              Condition:
                StringLike:
                  iam:PassedToService: "ec2.*"
```

Résolution des problèmes

Les tâches échouent avec iam :PassRole erreur d'autorisation

Si vos tâches échouent à cause d'un `ResourceInitializationError` qui mentionne `iam:PassRole`, vérifiez que le nom de votre rôle d'instance commence par `ecsInstanceRole`.

Vous pouvez vérifier le nom généré automatiquement dans CloudFormation console située sous votre console Ressources onglet. Si le nom ne correspond pas, soit :

- Ajouter `RoleName: ecsInstanceRole` à votre `AWS::IAM::Role` ressource.
- Ajouter un message explicite `iam:PassRole` politique intégrée à votre rôle dans l'infrastructure. Pour plus d'informations, consultez the section called “Option 2 : utiliser un nom de rôle personnalisé”.

Rôle IAM d’infrastructure Amazon ECS pour les équilibreurs de charge

Un rôle IAM d’infrastructure Amazon ECS pour les équilibreurs de charge permet à Amazon ECS de gérer les ressources des équilibreurs de charge dans vos clusters en votre nom. Il est utilisé dans les cas suivants :

- Vous souhaitez utiliser blue/green déploiements avec Amazon ECS. Le rôle d’infrastructure permet à Amazon ECS de gérer les ressources de l’équilibreur de charge pour vos déploiements.
- Vous avez besoin qu’Amazon ECS puisse créer, modifier ou supprimer des ressources d’équilibreur de charge telles que les groupes cibles et les écouteurs pendant les opérations de déploiement.

Lorsqu’Amazon ECS assume ce rôle pour prendre des mesures en votre nom, les événements seront visibles dans AWS CloudTrail. Si Amazon ECS utilise ce rôle pour gérer les ressources de l'équilibreur de charge pour votre blue/green déploiements, les CloudTrail bûche `roleSessionName` sera `ECSNetworkingWithELB` ou `ecs-service-scheduler`. Vous pouvez utiliser ce nom pour rechercher des événements dans CloudTrail console en filtrant pour Nom d'utilisateur.

Amazon ECS fournit une politique gérée qui contient les autorisations requises pour la gestion de l’équilibreur de charge. Pour plus d'informations, voir AmazonECSInfrastructureRolePolicyForLoadBalancers dans le AWS Guide de référence des politiques gérées.

Création du rôle d’infrastructure Amazon ECS pour les équilibreurs de charge

Remplacez tout *`user input`* avec vos propres informations.

1. Créez un fichier nommé `ecs-infrastructure-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "AllowAccessToECSForInfrastructureManagement",
      "Effect": "Allow",
      "Principal": {
        "Service": "ecs.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez ce qui suit AWS CLI commande pour créer un rôle nommé `ecsInfrastructureRoleForLoadBalancers` en utilisant la politique de confiance que vous avez créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --assume-role-policy-document file://ecs-infrastructure-trust-policy.json
```

3. Joignez le AWS géré `AmazonECSInfrastructureRolePolicyForLoadBalancers` politique à l'égard de `ecsInfrastructureRoleForLoadBalancers` rôle.

```
aws iam attach-role-policy \
      --role-name ecsInfrastructureRoleForLoadBalancers \
      --policy-arn arn:aws:iam::aws:policy/
AmazonECSInfrastructureRolePolicyForLoadBalancers
```

Vous pouvez également utiliser le flux de travail Stratégie d'approbation personnalisée de la console IAM pour créer le rôle. Pour plus d'informations, reportez-vous à la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le Guide de l'utilisateur IAM.

Important

Si le rôle d'infrastructure est utilisé par Amazon ECS pour gérer les ressources de l'équilibreur de charge pour votre blue/green déploiements, assurez-vous des points suivants avant de supprimer ou de modifier le rôle :

- Le rôle n'est pas supprimé lorsque des déploiements actifs sont en cours.
- La stratégie d'approbation associée au rôle n'est pas modifiée pour supprimer l'accès Amazon ECS (`ecs.amazonaws.com`).
- La politique gérée `AmazonECSInfrastructureRolePolicyForLoadBalancers` n'est pas supprimée lorsque des déploiements actifs sont en cours.

Supprimer ou modifier le rôle pendant qu'il est actif blue/green les déploiements peuvent entraîner des échecs de déploiement et laisser vos services dans un état incohérent.

Une fois le fichier créé, vous devez autoriser votre utilisateur à transmettre le rôle à Amazon ECS.

Autorisation de transmettre le rôle d'infrastructure à Amazon ECS

Pour utiliser un rôle IAM d'infrastructure ECS pour les équilibreurs de charge, vous devez autoriser votre utilisateur à transmettre le rôle à Amazon ECS. Attachez les autorisations `iam:PassRole` suivantes à votre utilisateur. Remplacer *`ecsInfrastructureRoleForLoadBalancers`* avec le nom du rôle d'infrastructure que vous avez créé.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": "iam:PassRole",
            "Effect": "Allow",
            "Resource":
 ["arn:aws:iam::*:role/ecsInfrastructureRoleForLoadBalancers"],
            "Condition": {
                "StringEquals": {"iam:PassedToService": "ecs.amazonaws.com"}
            }
        }
```

```
    ]
}
```

Pour plus d'informations sur `iam:Passrole` et pour mettre à jour les autorisations de votre utilisateur, consultez Accorder à un utilisateur l'autorisation de transmettre un rôle à un AWS service et Modifier les autorisations d'un utilisateur IAM dans le Gestion des identités et des accès AWS Guide de l'utilisateur.

Amazon ECS CodeDeploy Rôle IAM

Avant de pouvoir utiliser CodeDeploy blue/green type de déploiement avec Amazon ECS, le CodeDeploy le service a besoin d'autorisations pour mettre à jour votre service Amazon ECS en votre nom. Ces autorisations sont fournies par CodeDeploy Rôle IAM (`ecsCodeDeployRole`).

> **Note**
>
> Les utilisateurs ont également besoin d'autorisations pour utiliser CodeDeploy; ces autorisations sont décrites dans Autorisations IAM requises.

Deux politiques gérées sont fournies. Pour plus d'informations, consultez l'une des sections suivantes dans le Guide de référence des politiques gérées par AWS .

- AWSCodeDeployRoleForECS - donne CodeDeploy autorisation de mettre à jour n'importe quelle ressource à l'aide de l'action associée.
- AWSCodeDeployRoleForECSLimited - donne CodeDeploy autorisations plus limitées.

Création du CodeDeploy rôle

Vous pouvez utiliser les procédures suivantes pour créer un CodeDeploy rôle pour Amazon ECS

Console de gestion AWS

Pour créer le rôle de service pour CodeDeploy (console IAM)

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
2. Dans le volet de navigation de la console IAM, sélectionnez Rôles, puis Créer un rôle.

3. Pour Trusted entity (Entité de confiance), choisissez Service AWS.
4. Pour Service ou cas d'utilisation, choisissez CodeDeploy, puis choisissez CodeDeploy - ECOS cas d'utilisation.
5. Choisissez Suivant.
6. Dans le Joindre une politique d'autorisations section, assurez-vous que AWSCodeDeployRoleForECS la politique est sélectionnée.
7. Choisissez Suivant.
8. Pour Nom du rôle, entrez ecsCodeDeployRole.
9. Passez en revue les informations du rôle, puis choisissez Create role (Créer un rôle).

AWS CLI

Remplacez tout *user input* avec vos propres informations.

1. Créez un fichier nommé `codedeploy-trust-policy.json` qui contient la politique de confiance à utiliser pour CodeDeploy Rôle IAM.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "",
            "Effect": "Allow",
            "Principal": {
                "Service": ["codedeploy.amazonaws.com"]
            },
            "Action": "sts:AssumeRole"
        }
    ]
}
```

2. Créez un rôle IAM nommé `ecsCodedeployRole` à l'aide de la stratégie d'approbation créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsCodedeployRole \
```

```
      --assume-role-policy-document file://codedeploy-trust-policy.json
```

3. Attachez la politique gérée `AWSCodeDeployRoleForECS` ou `AWSCodeDeployRoleForECSLimited` au rôle `ecsTaskRole`.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECS
```

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam::aws:policy/AWSCodeDeployRoleForECSLimited
```

Lorsque les tâches de votre service nécessitent un rôle d'exécution de tâches, vous devez ajouter le `iam:PassRole` autorisation pour chaque rôle d'exécution de tâche ou remplacement du rôle de tâche par CodeDeploy rôle en tant que politique.

Autorisations du rôle d’exécution de tâche

Lorsque les tâches de votre service nécessitent un rôle d'exécution de tâches, vous devez ajouter le `iam:PassRole` autorisation pour chaque rôle d'exécution de tâche ou remplacement du rôle de tâche par CodeDeploy rôle en tant que politique. Pour plus d’informations, consultez Rôle IAM d'exécution de tâche Amazon ECS et rôle IAM de tâche Amazon ECS. Ensuite, vous associez cette politique au CodeDeploy rôle

Création de la stratégie

Console de gestion AWS

Pour utiliser l’éditeur de politique JSON afin de créer une politique

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
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 Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l’option JSON.

5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::<aws_account_id>:role/
<ecsCodeDeployRole>"]
        }
    ]
}
```

6. 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 plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez 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.
8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la politique, joignez-la au CodeDeploy rôle. Pour plus d'informations sur la manière d'associer une politique à un rôle, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .

AWS CLI

Remplacez tout *`user input`* avec vos propres informations.

1. Créez un fichier nommé `blue-green-iam-passrole.json` avec le contenu suivant.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::*:role/code-deploy-role"],
            "Condition": {
                    "StringEquals": {"iam:PassedToService":
 "ecs.amazonaws.com"}
            }
        }
    ]
}
```

2. Utilisez la commande suivante pour créer la politique IAM à l'aide du fichier de politique JSON.

```
aws iam create-policy \
      --policy-name cdTaskExecutionPolicy \
      --policy-document file://blue-green-iam-passrole.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`cdTaskExecutionPolicy`].Arn'
```

4. Utilisez la commande suivante pour associer la politique au CodeDeploy Rôle IAM.

```
aws iam attach-role-policy \
      --role-name ecsCodedeployRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/cdTaskExecutionPolicy
```

Amazon ECS EventBridge Rôle IAM

Avant de pouvoir utiliser les tâches planifiées d'Amazon ECS avec EventBridge règles et objectifs, les EventBridge le service a besoin d'autorisations pour exécuter les tâches Amazon ECS en votre nom. Ces autorisations proviennent du EventBridge Rôle IAM (`ecsEventsRole`).

Le `AmazonEC2ContainerServiceEventsRole` la politique est la suivante.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": ["ecs:RunTask"],
            "Resource": ["*"]
        },
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["*"],
            "Condition": {
                "StringLike": {"iam:PassedToService": "ecs-tasks.amazonaws.com"}
            }
        },
        {
            "Effect": "Allow",
            "Action": "ecs:TagResource",
            "Resource": "*",
            "Condition": {
                "StringEquals": {
                    "ecs:CreateAction": ["RunTask"]
                }
            }
        }
    ]
}
```

Si vos tâches planifiées nécessitent l'utilisation du rôle d'exécution des tâches, d'un rôle de tâche ou d'un remplacement de rôle de tâche, vous devez ajouter `iam:PassRole` les autorisations pour

chaque rôle d'exécution de tâche, rôle de tâche ou rôle de tâche sont remplacées par EventBridge Rôle IAM. Pour plus d'informations sur le rôle d'exécution de tâche, consultez Rôle IAM d'exécution de tâche Amazon ECS.

> **Note**
>
> Spécifiez l'ARN complet du rôle d'exécution de tâche ou du remplacement de rôle de tâche.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

Vous pouvez choisir de laisser Console de gestion AWS créer le EventBridge rôle qui vous est attribué lorsque vous configurez une tâche planifiée. Pour plus d'informations, consultez À l'aide d'Amazon EventBridge Planificateur pour planifier les tâches Amazon ECS.

Création du EventBridge rôle

Remplacez tout *`user input`* avec vos propres informations.

1. Créez un fichier nommé `eventbridge-trust-policy.json` contenant la stratégie d'approbation à utiliser pour le rôle IAM. Le fichier doit contenir ce qui suit :

   JSON

```
{
```

```
  "Version":"2012-10-17",
  "Statement": [
    {
      "Sid": "",
      "Effect": "Allow",
      "Principal": {
        "Service": "events.amazonaws.com"
      },
      "Action": "sts:AssumeRole"
    }
  ]
}
```

2. Utilisez la commande suivante pour créer un rôle IAM nommé `ecsEventsRole` à l'aide de la stratégie d'approbation que vous avez créée à l'étape précédente.

```
aws iam create-role \
      --role-name ecsEventsRole \
      --assume-role-policy-document file://eventbridge-trust-policy.json
```

3. Joignez le AWS géré `AmazonEC2ContainerServiceEventsRole` à la `ecsEventsRole` rôle à l'aide de la commande suivante.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam::aws:policy/service-role/
AmazonEC2ContainerServiceEventsRole
```

Vous pouvez également utiliser la console IAM Politique de confiance personnalisée flux de travail (https://console.aws.amazon.com/iam/) pour créer le rôle. Pour plus d'informations, reportez-vous à la section Création d'un rôle à l'aide de politiques de confiance personnalisées (console) dans le Guide de l'utilisateur IAM.

Joindre une politique au **`ecsEventsRole`** rôle

Vous pouvez utiliser les procédures suivantes pour ajouter des autorisations pour le rôle d'exécution des tâches dans EventBridge Rôle IAM.

Console de gestion AWS

Pour utiliser l'éditeur de politique JSON afin de créer une politique

1. Connectez-vous au Console de gestion AWS et ouvrez la console IAM à l'adresse https://console.aws.amazon.com/iam/.
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 Mise en route.
3. En haut de la page, sélectionnez Créer une politique.
4. Dans la section Éditeur de politique, choisissez l'option JSON.
5. Entrez le document de politique JSON suivant :

```
{
    "Version": "2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": ["arn:aws:iam::111122223333:role/
<ecsTaskExecutionRole_or_TaskRole_name>"]
        }
    ]
}
```

6. 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 plus d'informations, consultez la page Restructuration de politique dans le Guide de l'utilisateur IAM.

7. Sur la page Vérifier et créer, saisissez 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.

8. Choisissez Create policy (Créer une politique) pour enregistrer votre nouvelle politique.

Après avoir créé la politique, joignez-la au EventBridge rôle. Pour plus d'informations sur la manière d'associer une politique à un rôle, consultez la section Mise à jour des autorisations pour un rôle dans le Guide de l'utilisateur Gestion des identités et des accès AWS .

AWS CLI

Remplacez tout *user input* avec vos propres informations.

1. Créez un fichier nommé `ev-iam-passrole.json` avec le contenu suivant.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": "iam:PassRole",
            "Resource": [

 "arn:aws:iam::111122223333:role/ecsTaskExecutionRole_or_TaskRole_name"
            ]
        }
    ]
}
```

2. Utilisez ce qui suit AWS CLI commande pour créer la politique IAM à l'aide du fichier de document de stratégie JSON.

```
aws iam create-policy \
      --policy-name eventsTaskExecutionPolicy \
      --policy-document file://ev-iam-passrole.json
```

3. Récupérez l'ARN de la politique IAM que vous avez créée à l'aide de la commande suivante.

```
aws iam list-policies --scope Local --query 'Policies[?
PolicyName==`eventsTaskExecutionPolicy`].Arn'
```

4. Utilisez la commande suivante pour joindre la politique au EventBridge Rôle IAM à l'aide de l'ARN de stratégie.

```
aws iam attach-role-policy \
      --role-name ecsEventsRole \
      --policy-arn arn:aws:iam:111122223333:aws:policy/eventsTaskExecutionPolicy
```

Autorisations requises pour la console Amazon ECS

Conformément à la bonne pratique pour accorder le moindre privilège, vous pouvez utiliser la stratégie gérée par `AmazonECS_FullAccess` comme modèle pour créer votre propre stratégie personnalisée. De cette façon, vous pouvez supprimer ou ajouter des autorisations pour la stratégie gérée en fonction de vos besoins spécifiques. Pour plus d'informations, voir Amazon ECS_FullAccess dans le AWS Référence des politiques gérées.

Autorisations pour créer des rôles IAM

Les actions suivantes nécessitent des autorisations supplémentaires pour terminer l'opération :

- Enregistrement d'une instance externe : pour plus d'informations, veuillez consulter Rôle IAM Amazon ECS Anywhere.
- Enregistrement d'une définition de tâche : pour plus d'informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.
- Création d'un EventBridge règle à utiliser pour planifier les tâches - pour plus d'informations, voir Amazon ECS EventBridge Rôle IAM

Vous pouvez ajouter ces autorisations en créant un rôle dans IAM avant de les utiliser dans la console Amazon ECS. Si vous ne créez pas les rôles, la console Amazon ECS les crée en votre nom.

Autorisations requises pour enregistrer une instance externe dans un cluster

Vous avez besoin d'autorisations supplémentaires lorsque vous enregistrez une instance externe dans un cluster et que vous souhaitez créer un nouveau rôle d'instance externe (`ecsExternalInstanceRole`).

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et répertorier les rôles IAM et les politiques qui leur sont attachées.
  - iam :AttachRolePolicy
  - iam :CreateRole
  - suis :CreateInstanceProfile
  - iam :AddRoleToInstanceProfile
  - iam :ListInstanceProfilesForRole
  - iam :GetRole
- `ssm` : permet aux principaux d'enregistrer l'instance externe auprès de Systems Manager.

> **Note**
>
> Pour choisir un `ecsExternalInstanceRole` existant, vous devez disposer des autorisations `iam:GetRole` et `iam:PassRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsExternalInstanceRole`.

JSON

```
{
"Statement": [
      {
          "Effect": "Allow",
          "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
          ],
          "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
      },
      {
          "Effect": "Allow",
```

```
            "Action": ["iam:PassRole","ssm:CreateActivation"],
            "Resource": "arn:aws:iam::*:role/ecsExternalInstanceRole"
        }
    ]
}
```

Autorisations requises pour enregistrer une définition de tâche

Vous avez besoin d'autorisations supplémentaires lorsque vous enregistrez une définition de tâche et que vous souhaitez créer un nouveau rôle d'exécution de tâche (`ecsTaskExecutionRole`).

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et répertorier les rôles IAM et les politiques qui leur sont attachées.
  - iam :AttachRolePolicy
  - iam :CreateRole
  - iam :GetRole

> **Note**
>
> Pour choisir un `ecsTaskExecutionRole` existant, vous devez disposer de l'autorisation `iam:GetRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsTaskExecutionRole`.

JSON

```
{
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "iam:AttachRolePolicy",
            "iam:CreateRole",
            "iam:GetRole"
```

```
        ],
        "Resource": "arn:aws:iam::*:role/ecsTaskExecutionRole"
        }
    ]
}
```

Autorisations requises pour utiliser Amazon Q Developer afin de fournir des recommandations dans la console

Pour qu'Amazon Q Developer puisse formuler des recommandations dans la console ECS, vous devez activer les autorisations IAM adéquates pour votre rôle ou utilisateur IAM. Vous devez ajouter l'autorisation `codewhisperer:GenerateRecommendations`.

JSON

```
{
"Statement": [
      {
            "Sid": "AmazonQDeveloperPermissions",
            "Effect": "Allow",
            "Action": ["codewhisperer:GenerateRecommendations"],
            "Resource": "*"
        }
    ]
}
```

Pour utiliser le chat en ligne dans la console Amazon ECS, vous devez activer les autorisations IAM adéquates pour votre rôle ou utilisateur IAM. Vous devez ajouter l'autorisation `q:SendMessage` :

JSON

```
{
"Statement": [
    {
        "Sid": "AmazonQDeveloperInlineChatPermissions",
        "Effect": "Allow",
        "Action": ["q:SendMessage"],
        "Resource": "*"
```

```
    }
  ]
}
```

Autorisations requises pour créer un EventBridge règle pour les tâches planifiées

Vous avez besoin d'autorisations supplémentaires lorsque vous planifiez une tâche et que vous souhaitez en créer une nouvelle CloudWatch Rôle des événements (`ecsEventsRole`) rôle.

Les autorisations supplémentaires suivantes sont requises :

- `iam` : permet aux principaux de créer et de répertorier les rôles IAM et les politiques qui leur sont associées, et d'autoriser Amazon ECS à transmettre le rôle à d'autres services pour qu'ils l'endossent.

> **Note**
>
> Pour choisir un `ecsEventsRole` existant, vous devez disposer des autorisations `iam:GetRole` et `iam:PassRole`.

La politique suivante contient les autorisations requises et limite les actions au rôle `ecsEventsRole`.

JSON

```
{
 "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "iam:AttachRolePolicy",
            "iam:CreateRole",
            "iam:GetRole",
            "iam:PassRole"
        ],
        "Resource": "arn:aws:iam::*:role/ecsEventsRole"
        }
    ]
}
```

Autorisations nécessaires pour l’affichage des déploiements de service

Lorsque vous suivez la pratique exemplaire consistant à octroyer le moindre privilège, vous devez ajouter des autorisations supplémentaires afin de visualiser les déploiements de service dans la console.

Vous devez avoir accès aux actions suivantes :

- ListServiceDeployments
- DescribeServiceDeployments
- DescribeServiceRevisions

Vous devez avoir accès aux ressources suivantes :

- Service
- Déploiements de service
- Révision de service

L’exemple de politique suivant contient les autorisations requises et limite les actions à un service spécifié.

Remplacez `account`, `cluster-name` et `service-name` par vos propres valeurs.

JSON

```
{
"Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "ecs:ListServiceDeployments",
            "ecs:DescribeServiceDeployments",
            "ecs:DescribeServiceRevisions"
        ],
        "Resource": [
            "arn:aws:ecs:us-east-1:123456789012:service/cluster-name/service-
name",
            "arn:aws:ecs:us-east-1:123456789012:service-deployment/cluster-name/
service-name/*",
```

```
            "arn:aws:ecs:us-east-1:123456789012:service-revision/cluster-name/
service-name/*"
            ]
        }
    ]
}
```

Autorisations requises pour afficher les événements du cycle de vie Amazon ECS dans Container Insights

Les autorisations suivantes sont requises pour afficher les événements du cycle de vie. Ajoutez les autorisations suivantes sous forme de politique en ligne au rôle. Pour plus d'informations, consultez la section Ajout et suppression de politiques IAM.

- événements :DescribeRule
- événements :ListTargetsByRule
- journaux :DescribeLogGroups

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:DescribeRule",
        "events:ListTargetsByRule",
        "logs:DescribeLogGroups"
      ],
      "Resource": "*"
    }
  ]
}
```

Autorisations requises pour l’affichage des événements du cycle de vie Amazon ECS dans Container Insights

Les autorisations suivantes sont requises pour configurer les événements du cycle de vie :

- événements :PutRule
- événements :PutTargets
- journaux :CreateLogGroup

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "events:PutRule",
        "events:PutTargets",
        "logs:CreateLogGroup"
      ],
      "Resource": "*"
    }
  ]
}
```

Autorisations requises pour la console Amazon ECS avec CloudFormation

Avant d'utiliser Console de gestion AWS pour créer vos ressources, vous devez vous assurer de disposer des autorisations IAM appropriées. Pour plus d’informations sur la façon de configurer les autorisations pour la console Amazon ECS en général, consultez d’abord la section Autorisations requises pour la console Amazon ECS.

La console Amazon ECS est alimentée par AWS CloudFormation et nécessite des autorisations IAM supplémentaires dans les cas suivants :

- Création d’un cluster
- Création d’un service

- Création d'un fournisseur de capacité

Vous pouvez créer une politique pour les autorisations supplémentaires, puis les associer au rôle IAM que vous utilisez pour accéder à la console. Pour plus d'informations, consultez Création de politiques IAM dans le Guide de l'utilisateur IAM.

Autorisations requises pour créer un cluster

Lorsque vous créez un cluster dans la console, vous avez besoin d'autorisations supplémentaires qui vous permettent de gérer CloudFormation piles.

Les autorisations supplémentaires suivantes sont requises :

- `cloudformation` : permet aux mandataires de créer et de gérer des piles CloudFormation . Ceci est requis lors de la création de clusters Amazon ECS à l'aide de la Console de gestion AWS et pour la gestion ultérieure de ces groupes.
- `ssm` — Permet CloudFormation pour faire référence à la dernière version d'Amazon ECS-optimized AMI. Cela est obligatoire lors de la création de clusters Amazon ECS à l'aide du Console de gestion AWS.

La politique suivante contient les éléments requis CloudFormation autorisations et limite les actions aux ressources créées dans la console Amazon ECS.

JSON

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
            ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/Infra-ECS-Cluster-*"
            ]
      },
```

```
      {
          "Effect": "Allow",
          "Action": "ssm:GetParameters",
          "Resource": [
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-
linux-2*/*",
            "arn:aws:ssm:*:*:parameter/aws/service/ecs/optimized-ami/amazon-
linux-2023*/*"
          ]
      }
   ]
}
```

Si vous n'avez pas créé le rôle d'instance de conteneur Amazon ECS (`ecsInstanceRole`) et que vous créez un cluster qui utilise des instances Amazon EC2, la console créera le rôle en votre nom.

En outre, si vous utilisez des groupes Auto Scaling, vous avez besoin d’autorisations supplémentaires pour que la console puisse ajouter des balises aux groupes Auto Scaling lors de l’utilisation de la fonctionnalité d’autoscaling de cluster.

Les autorisations supplémentaires suivantes sont requises :

- `autoscaling` : permet à la console de baliser le groupe Amazon EC2 Auto Scaling. Ceci est requis pour la gestion des groupes Amazon EC2 Auto Scaling lorsque vous utilisez la fonction de scalabilité automatique des clusters (Auto Scaling). Le tag est ECS-managed balise que la console ajoute automatiquement au groupe pour indiquer qu'elle a été créée dans la console.
- `iam` : permet aux mandataires de répertorier les rôles IAM et les stratégies qui leur sont attachées. Les mandataires peuvent également répertorier les profils d'instance disponibles pour vos instances Amazon EC2.

La politique suivante contient les autorisations IAM requises et limite les actions au rôle `ecsInstanceRole`.

Les autorisations Auto Scaling ne sont pas limitées.

JSON

```
{
  "Statement": [
```

```
      {
          "Effect": "Allow",
            "Action": [
              "iam:AttachRolePolicy",
              "iam:CreateRole",
              "iam:CreateInstanceProfile",
              "iam:AddRoleToInstanceProfile",
              "iam:ListInstanceProfilesForRole",
              "iam:GetRole"
            ],
            "Resource": "arn:aws:iam::*:role/ecsInstanceRole"
        },
        {
            "Effect": "Allow",
            "Action": "autoscaling:CreateOrUpdateTags",
            "Resource": "*"
        }
    ]
}
```

Autorisations requises pour créer un service

Lorsque vous créez un service dans la console, vous avez besoin d'autorisations supplémentaires qui vous permettent de gérer CloudFormation piles. Les autorisations supplémentaires suivantes sont requises :

- `cloudformation` : permet aux mandataires de créer et de gérer des piles CloudFormation . Cela est requis lors de la création de services Amazon ECS à l'aide de l' Console de gestion AWS et pour la gestion ultérieure de ces groupes.

La politique suivante contient les autorisations requises et limite les actions aux ressources créées dans la console Amazon ECS.

JSON

```
{
  "Statement": [
      {
          "Effect": "Allow",
          "Action": [
```

```
                "cloudformation:CreateStack",
                "cloudformation:DeleteStack",
                "cloudformation:DescribeStack*",
                "cloudformation:UpdateStack"
            ],
            "Resource": [
                "arn:*:cloudformation:*:*:stack/ECS-Console-V2-Service-*"
            ]
        }
    ]
}
```

Autorisations requises pour les fonctions Lambda dans Amazon ECS blue/green déploiements

Lorsque vous utilisez les fonctions Lambda comme crochets du cycle de vie du déploiement dans Amazon ECS blue/green déploiements, vous devez créer un rôle IAM avec des autorisations spécifiques. Ce rôle autorise Amazon ECS à invoquer vos fonctions Lambda à différentes étapes du cycle de vie du déploiement.

Les autorisations supplémentaires suivantes sont requises :

- `lambda:InvokeFunction` : permet à Amazon ECS d'invoquer des fonctions Lambda configurées comme des hooks de cycle de vie pendant le processus de déploiement.

Pour la stratégie d'approbation, vous devez autoriser le service à assumer ce rôle :

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Principal": {
                "Service": [
                    "ecs.amazonaws.com"
                ]
            },
```

```
            "Action": "sts:AssumeRole"
        }
    ]
}
```

Autorisations IAM requises pour l'autoscaling d'un service Amazon ECS

Service Auto Scaling est rendu possible par une combinaison d'Amazon ECS, CloudWatchet des API Application Auto Scaling. Les services sont créés et mis à jour avec Amazon ECS, les alarmes sont créées avec CloudWatch, et les politiques de dimensionnement sont créées avec Application Auto Scaling.

Outre les autorisations IAM standard pour la création et la mise à jour de services, les autorisations suivantes sont requises pour interagir avec les paramètres de Service Auto Scaling.

- `application-autoscaling:*`
- `ecs:DescribeServices`
- `ecs:UpdateService`
- `cloudwatch:DescribeAlarms`
- `cloudwatch:PutMetricAlarm`
- `cloudwatch:DeleteAlarms`
- `cloudwatch:DescribeAlarmHistory`
- `cloudwatch:DescribeAlarmsForMetric`
- `cloudwatch:GetMetricStatistics`
- `cloudwatch:ListMetrics`
- `cloudwatch:DisableAlarmActions`
- `cloudwatch:EnableAlarmActions`
- `iam:CreateServiceLinkedRole`
- `sns:CreateTopic`
- `sns:Subscribe`
- `sns:Get*`
- `sns:List*`

Le service Application Auto Scaling a également besoin d'une autorisation pour décrire vos services Amazon ECS et CloudWatch alarmes et autorisations pour modifier le nombre souhaité de votre service en votre nom. Le `sns:` les autorisations concernent les notifications qui CloudWatch envoie à une rubrique Amazon SNS lorsqu'un seuil a été dépassé. Si vous utilisez la scalabilité automatique pour vos services Amazon ECS services, elle crée un rôle lié à un service nommé `AWSServiceRoleForApplicationAutoScaling_ECSService`. Ce rôle lié à un service accorde l'autorisation Application Auto Scaling de décrire les alarmes de vos politiques, de contrôler le nombre de tâches en cours d'exécution du service et de modifier le nombre souhaité du service. Le rôle d'origine géré par Amazon ECS pour Application Auto Scaling était `ecsAutoscaleRole`, mais il n'est plus nécessaire. Le rôle lié à un service est le rôle par défaut pour Application Auto Scaling. Pour plus d'informations, voir Service-linked rôles pour Application Auto Scaling dans le Guide de l'utilisateur de l'application Auto Scaling.

Si vous avez créé votre rôle d'instance de conteneur Amazon ECS auparavant CloudWatch des métriques sont disponibles pour Amazon ECS, vous devrez peut-être ajouter `ecs:StartTelemetrySession` autorisation. Pour plus d’informations, consultez Considérations.

Octroi de l'autorisation de baliser les ressources lors de la création

Les actions de balisage lors de la création d'API Amazon ECS suivantes vous permettent de spécifier des balises lorsque vous créez la ressource. Si des balises sont spécifiées dans l'action de création de ressources, AWS effectue une autorisation supplémentaire pour vérifier que les autorisations appropriées sont attribuées pour créer des balises.

- `CreateCapacityProvider`
- `CreateCluster`
- `CreateService`
- `CreateTaskSet`
- `RegisterContainerInstance`
- `RegisterTaskDefinition`
- `RunTask`
- `StartTask`

Vous pouvez utiliser des balises de ressource pour implémenter le contrôle basé sur les attributs (ABAC). Pour plus d’informations, consultez the section called “Contrôle de l'accès aux ressources Amazon ECS à l'aide des balises de ressources” et Balisage de ressources .

Pour autoriser le balisage lors de la création, créez ou modifiez une politique afin d'inclure à la fois les autorisations d'utilisation de l'action qui crée la ressource, telle que `ecs:CreateCluster` ou `ecs:RunTask` et l'action `ecs:TagResource`.

L'exemple suivant illustre une politique qui autorise les utilisateurs à créer des clusters et à ajouter des balises lors de la création du cluster. Les utilisateurs ne sont pas autorisés à attribuer des balises aux ressources existantes (ils ne peuvent pas appeler l'action `ecs:TagResource` directement).

```
{
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
         "ecs:CreateCluster"
      ],
      "Resource": "*"
    },
    {
      "Effect": "Allow",
      "Action": [
         "ecs:TagResource"
      ],
      "Resource": "*",
      "Condition": {
         "StringEquals": {
                  "ecs:CreateAction": [
                      "CreateCluster",
                      "CreateCapacityProvider",
                      "CreateService",
                      "CreateTaskSet",
                      "RegisterContainerInstance",
                      "RegisterTaskDefinition",
                      "RunTask",
                      "StartTask"
                  ]
          }
       }
    }
  ]
}
```

L'action `ecs:TagResource` est uniquement évaluée si les balises sont appliquées pendant l'action de création de ressources. Par conséquent, un utilisateur qui est autorisé à créer une ressource (en supposant qu'il n'existe aucune condition de balisage) n'a pas besoin des autorisations d'utiliser l'action `ecs:TagResource` si aucune balise n'est spécifié dans la demande. Toutefois, si l'utilisateur essaie de créer une ressource avec des balises, la demande échoue s'il n'a pas les autorisations d'utiliser l'action `ecs:TagResource`.

Contrôle de l'accès à des balises spécifiques dans Amazon ECS

Vous pouvez utiliser des conditions supplémentaires dans l'élément `Condition` de vos stratégies IAM pour contrôler les clés de balise et les valeurs qui peuvent être appliquées aux ressources.

Les clés de condition suivantes peuvent être utilisées avec les exemples de la section précédente :

- `aws:RequestTag` : Pour indiquer qu'une clé de balise ou une clé et valeur de balise particulière doit être présente dans une demande. D'autres balises peuvent également être spécifiées dans la demande.
  - Utilisez avec l'opérateur de condition `StringEquals` pour appliquer une combinaison de clé de balise et de valeur spécifique ; par exemple, pour appliquer la balise `cost-center=cc123` :

    ```
    "StringEquals": { "aws:RequestTag/cost-center": "cc123" }
    ```

  - A utiliser avec l'opération de condition `StringLike` pour appliquer une clé de balise spécifique dans la demande ; par exemple, pour appliquer la clé de balise `purpose`:

    ```
    "StringLike": { "aws:RequestTag/purpose": "*" }
    ```

- `aws:TagKeys` : Pour appliquer les clés de balise qui sont utilisées dans la demande.
  - A utiliser avec le modificateur `ForAllValues` pour appliquer des clés de balise spécifiques si celles-ci sont fournies dans la demande (si les balises sont spécifiées dans la demande, seules les clés de balise spécifiques sont autorisées ; aucune autre balise n'est autorisée). Par exemple, les clés de balise `environment` ou `cost-center` sont autorisées :

    ```
    "ForAllValues:StringEquals": { "aws:TagKeys": ["environment","cost-center"] }
    ```

  - A utiliser avec le modificateur `ForAnyValue` pour appliquer la présence d'au moins l'une des clés de balise spécifiées dans la demande. Par exemple, au moins l'une des clés de balise `environment` ou `webserver` doit être présente dans la demande :

```
"ForAnyValue:StringEquals": { "aws:TagKeys": ["environment","webserver"] }
```

Ces clés de condition peuvent être appliquées aux actions de création de ressources qui prennent en charge le balisage, ainsi qu'à l'action `ecs:TagResource`. Pour savoir si une action d'API Amazon ECS prend en charge le balisage, veuillez consulter Actions, ressources et clés de condition pour Amazon ECS.

Pour forcer les utilisateurs à spécifier des balises quand ils créent une ressource, vous devez utiliser la clé de condition `aws:RequestTag` ou la clé de condition `aws:TagKeys` avec le modificateur `ForAnyValue` sur l'action de création de ressources. L'action `ecs:TagResource` n'est pas évaluée si un utilisateur ne spécifie pas de balises pour l'action de création de ressources.

Pour les conditions, la clé de condition n'est pas sensible à la casse et la valeur de la condition est sensible à la casse. Par conséquent pour forcer la sensibilité à la casse d'une clé de balise, utilisez la clé de condition `aws:TagKeys`, où la clé de balise est indiquée comme une valeur dans la condition.

Pour plus d'informations sur les conditions à valeur multiples, consultez la section Conditions avec plusieurs clés ou valeurs de contexte dans le Guide de l'utilisateur IAM.

Contrôle de l'accès aux ressources Amazon ECS à l'aide des balises de ressources

Lorsque vous créez une politique IAM qui accorde aux utilisateurs l'autorisation d'utiliser les ressources Amazon ECS, vous pouvez inclure des informations de balise dans l'élément `Condition` de la politique pour contrôler l'accès en fonction des balises. Ceci est connu sous le nom de contrôle d'accès basé sur les attributs (ABAC). ABAC vous offre un meilleur contrôle sur les ressources qu'un utilisateur peut modifier, utiliser ou supprimer. Pour plus d'informations, consultez Présentation d'ABAC pour AWS.

Par exemple, vous pouvez créer une politique qui permet aux utilisateurs de supprimer un cluster, mais qui refuse l'action si le cluster possède la balise `environment=production`. Pour ce faire, vous utilisez la clé de condition `aws:ResourceTag` pour autoriser ou refuser l'accès à la ressource en fonction des balises attachées à la ressource.

```
"StringEquals": { "aws:ResourceTag/environment": "production" }
```

Pour savoir si une action d'API Amazon ECS prend en charge le contrôle d'accès à l'aide de la clé de condition `aws:ResourceTag`, veuillez consulter Actions, ressources et clés de condition pour

Amazon ECS. Notez que les actions `Describe` ne prennent pas en charge les autorisations au niveau des ressources, vous devez donc les spécifier dans une instruction distincte sans condition.

Par exemple les stratégies IAM, consultez Exemples de politiques Amazon ECS.

Si vous autorisez ou refusez à des utilisateurs l'accès à des ressources en fonction de balises, vous devez envisager de refuser de manière explicite la possibilité pour les utilisateurs d'ajouter ces balises ou de les supprimer des mêmes ressources. Sinon, il sera possible pour un utilisateur de contourner vos restrictions et d'obtenir l'accès à une ressource en modifiant ses balises.

Exemples de politiques Amazon ECS

Vous pouvez utiliser les politiques IAM pour octroyer aux utilisateurs les autorisations d'afficher et d'utiliser des ressources spécifiques sur la console Amazon ECS. Vous pouvez utiliser les exemples de politiques de la section précédente ; toutefois, ils sont conçus pour les demandes effectuées avec AWS CLI ou un AWS SDK.

Exemple : autorisation de suppression d'un cluster Amazon ECS par les utilisateurs en fonction des balises

La politique suivante permet aux utilisateurs de supprimer des clusters lorsque la balise possède un key/value paire de »Purpose/Testing«.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Action": [
                "ecs:DeleteCluster"
            ],
            "Effect": "Allow",
            "Resource": "arn:aws:ecs:us-east-1:111122223333:cluster/*",
            "Condition": {
                "StringEquals": {
                    "aws:ResourceTag/Purpose": "Testing"
                }
            }
        }
    ]
```

```
}
```

Dépannage de l'identité et de l'accès Amazon Elastic Container Service

Utilisez les informations suivantes pour identifier et résoudre les problèmes courants que vous pouvez rencontrer lorsque vous travaillez avec Amazon ECS et IAM.

Rubriques

- Je ne suis pas autorisé à effectuer une action dans Amazon ECS
- Je ne suis pas autorisé à effectuer iam :PassRole
- Je souhaite autoriser des personnes extérieures à mon Compte AWS pour accéder à mes ressources Amazon ECS
- Je rencontre des problèmes avec mon profil d'instance Amazon ECS Managed Instances
- Ressources supplémentaires pour la résolution des problèmes

Je ne suis pas autorisé à effectuer une action dans Amazon ECS

Si vous recevez une erreur qui indique que vous n'êtes pas autorisé à effectuer une action, vos politiques doivent être mises à jour afin de vous permettre d'effectuer l'action.

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 `ecs:`*`GetWidget`* fictives.

```
User: arn:aws:iam::123456789012:user/mateojackson is not authorized to perform:
 ecs:GetWidget on resource: my-example-widget
```

Dans ce cas, la politique qui s'applique à l'utilisateur `mateojackson` doit être mise à jour pour autoriser l'accès à la ressource *`my-example-widget`* à l'aide de l'action `ecs:`*`GetWidget`*.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Je ne suis pas autorisé à effectuer iam :PassRole

Si vous recevez une erreur selon laquelle vous n'êtes pas autorisé à exécuter l'action `iam:PassRole`, vos politiques doivent être mises à jour pour vous permettre de transmettre un rôle à Amazon ECS.

Certains Services AWS vous permettent de transmettre un rôle existant à ce service au lieu de créer un nouveau rôle de service ou un rôle lié à un service. Pour ce faire, vous devez disposer des autorisations nécessaires pour transmettre le rôle au service.

L'exemple d'erreur suivant se produit lorsqu'un utilisateur IAM nommé `marymajor` essaie d'utiliser la console pour effectuer une action dans Amazon ECS. Toutefois, l'action nécessite que le service ait des autorisations accordées par un rôle de service. Mary n'est pas autorisée à transmettre le rôle au service.

```
User: arn:aws:iam::123456789012:user/marymajor is not authorized to perform:
 iam:PassRole
```

Dans ce cas, les politiques de Mary doivent être mises à jour pour lui permettre d'exécuter l'action `iam:PassRole`.

Si vous avez besoin d'aide, contactez votre AWS administrateur. Votre administrateur vous a fourni vos informations d'identification de connexion.

Si vous utilisez des instances gérées Amazon ECS et que vous recevez cette erreur, le nom de votre rôle d'instance peut ne pas correspondre à la convention de dénomination requise par la politique gérée. Pour plus d'informations, consultez Je rencontre des problèmes avec mon profil d'instance Amazon ECS Managed Instances.

Je souhaite autoriser des personnes extérieures à mon Compte AWS pour accéder à mes ressources Amazon ECS

Vous pouvez créer un rôle que les utilisateurs provenant d'autres comptes ou les personnes extérieures à votre organisation pourront utiliser pour accéder à vos ressources. Vous pouvez spécifier qui est autorisé à assumer le rôle. Pour les services qui prennent en charge les politiques basées sur les ressources ou les listes de contrôle d'accès (ACL), vous pouvez utiliser ces politiques pour donner l'accès à vos ressources.

Pour plus d'informations, consultez les éléments suivants :

- Pour savoir si Amazon ECS est compatible avec ces fonctions, veuillez consulter Fonctionnement d'Amazon Elastic Container Service avec IAM.
- Pour savoir comment donner accès à vos ressources sur Comptes AWS que vous possédez, voyez Fournir l'accès à un utilisateur IAM dans un autre Compte AWS que tu possèdes dans le Guide de l'utilisateur IAM.
- Pour savoir comment donner accès à vos ressources à des tiers Comptes AWS, voir Fournir un accès à Comptes AWS appartenant à des tiers dans le Guide de l'utilisateur IAM.
- Pour savoir comment fournir un accès par le biais de la fédération d'identité, consultez Fournir un accès à des utilisateurs authentifiés en externe (fédération d'identité) dans le Guide de l'utilisateur IAM.
- Pour en savoir plus sur la différence entre l'utilisation des rôles et des politiques basées sur les ressources pour l'accès intercompte, consultez Accès intercompte aux ressources dans IAM dans le Guide de l'utilisateur IAM.

Je rencontre des problèmes avec mon profil d'instance Amazon ECS Managed Instances

Si vous utilisez `AmazonECSInfrastructureRolePolicyForManagedInstances` politique gérée, le nom du rôle d'instance doit commencer par `ecsInstanceRole`. Les champs d'application de la politique `iam:PassRole` pour `arn:aws:iam::*:role/ecsInstanceRole*`, donc un nom différent provoque une erreur d'autorisation lors du lancement de la tâche.

Ceci est courant avec CloudFormation lorsque vous omettez `RoleName` à partir de votre `AWS::IAM::Role` ressource, car CloudFormation génère automatiquement des noms tels que `MyStack-InstanceRole-ABC123` qui ne correspondent pas à la condition de la politique.

Pour résoudre ce problème, procédez de l'une des manières suivantes :

- Ajouter `RoleName: ecsInstanceRole` à votre `AWS::IAM::Role` ressource afin que le nom corresponde à la politique gérée.
- Ajouter un message explicite `iam:PassRole` politique intégrée à votre rôle d'infrastructure qui cible l'ARN du rôle d'instance.

Pour CloudFormation modèles et étapes détaillées, voir Créez le profil d'instance à l'aide de CloudFormation.

Ressources supplémentaires pour la résolution des problèmes

Les pages suivantes fournissent des informations sur les codes d'erreur :

- Messages d'erreurs liées aux tâches Amazon ECS arrêtées
- Affichage des messages d'événement d'un service Amazon ECS

Pratiques exemplaires relatives à IAM pour Amazon ECS

Vous pouvez utiliser Gestion des identités et des accès AWS (IAM) pour gérer et contrôler l'accès à votre AWS services et ressources via des politiques basées sur des règles à des fins d'authentification et d'autorisation. Plus précisément, grâce à ce service, vous contrôlez l'accès à votre AWS ressources en utilisant des politiques appliquées aux utilisateurs, aux groupes ou aux rôles. Parmi ces trois, les utilisateurs sont des comptes qui peuvent avoir accès à vos ressources. De plus, un rôle IAM est un ensemble d'autorisations qui peuvent être endossées par une identité authentifiée, qui n'est pas associée à une identité particulière en dehors d'IAM. Pour plus d'informations, consultez la section Présentation d'Amazon ECS sur la gestion des accès : autorisations et politiques.

Respect de la politique de moindre privilège d'accès

Créez des politiques dont le champ d'application est défini pour permettre aux utilisateurs d'effectuer les tâches qui leur sont prescrites. Par exemple, si un développeur doit régulièrement arrêter une tâche, créez une politique qui autorise uniquement cette action particulière. L'exemple suivant permet uniquement à un utilisateur d'arrêter une tâche qui appartient à une `task_family` spécifique dans un cluster avec un Amazon Resource Name (ARN) spécifique. La référence à un ARN dans une condition est également un exemple d'utilisation d'autorisations au niveau des ressources. Vous pouvez utiliser les autorisations au niveau des ressources pour spécifier la ressource à laquelle vous souhaitez qu'une action s'applique. Pour plus d'informations, voir Ressources relatives aux politiques pour Amazon ECS.

Utilisation des ressources du cluster comme limite administrative

Les politiques dont le champ d'application est trop restreint peuvent entraîner une prolifération de rôles et augmenter les frais administratifs. Plutôt que de créer des rôles limités à des tâches ou à des services spécifiques uniquement, créez des rôles délimités à des clusters et utilisez le cluster comme limite administrative principale.

Création de pipelines automatisés pour isoler les utilisateurs finaux de l’API

Vous pouvez limiter les actions que les utilisateurs peuvent utiliser en créant des pipelines qui empaquettent et déploient automatiquement des applications sur des clusters Amazon ECS. Cela délègue efficacement la création, la mise à jour et la suppression de tâches au pipeline. Pour plus d'informations, voir Tutoriel : déploiement standard d'Amazon ECS avec CodePipeline dans le AWS CodePipeline Guide de l'utilisateur.

Utilisez les conditions de politiques pour renforcer la couche de sécurité

Lorsque vous avez besoin d'un niveau de sécurité supplémentaire, ajoutez une condition à votre politique. Cela peut être utile si vous effectuez une opération privilégiée ou lorsque vous devez restreindre l'ensemble des actions pouvant être effectuées sur des ressources spécifiques. L'exemple de politique suivant nécessite une autorisation multifactorielle lors de la suppression d'un cluster.

JSON

```
{
  "Version":"2012-10-17",
  "Statement": [
    {
      "Effect": "Allow",
      "Action": [
        "ecs:DeleteCluster"
      ],
      "Condition": {
        "Bool": {
          "aws:MultiFactorAuthPresent": "true"
        }
      },
    "Resource": ["*"]
    }
  ]
}
```

Les balises appliquées aux services sont propagées à toutes les tâches qui font partie de ce service. De ce fait, vous pouvez créer des rôles limités aux ressources Amazon ECS avec des balises spécifiques. Dans la politique suivante, un principal IAM démarre et arrête toutes les tâches avec une clé de balise `Department` et une valeur de balise de `Accounting`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ecs:StartTask",
                "ecs:StopTask",
                "ecs:RunTask"
            ],
            "Resource": "arn:aws:ecs:us-east-1:123456789012:cluster/my-cluster",
            "Condition": {
                "StringEquals": {"ecs:ResourceTag/Department": "Accounting"}
            }
        }
    ]
}
```

Audit régulier de l’accès aux API

Un utilisateur peut changer de rôle. Une fois son rôle changé, les autorisations qui lui ont été accordées précédemment risquent de ne plus s'appliquer. Assurez-vous de vérifier qui a accès aux API Amazon ECS et de vérifier si cet accès est toujours justifié. Envisagez d'intégrer IAM à une solution de gestion du cycle de vie des utilisateurs qui révoque automatiquement l'accès lorsqu'un utilisateur quitte l'organisation. Pour plus d’informations, consultez la section Consignes pour les audits de sécuritéAWS dans le Guide de l’utilisateur Gestion des identités et des accès AWS .

Journalisation et surveillance dans Amazon Elastic Container Service

La surveillance joue un rôle important dans le maintien de la fiabilité, de la disponibilité et des performances d'Amazon Elastic Container Service et de votre AWS solutions. Vous devez collecter des données de surveillance provenant de toutes les parties de votre AWS solution qui vous permet de déboguer plus facilement une panne multipoint si elle survient. AWS fournit plusieurs outils pour surveiller vos ressources Amazon ECS et répondre aux incidents potentiels :

Amazon CloudWatch Alarmes

Surveillez une seule métrique sur une durée définie et exécutez une ou plusieurs actions en fonction de la valeur de la métrique par rapport à un seuil donné sur un certain nombre de durées. L'action est une notification envoyée à une rubrique Amazon Simple Notification Service (Amazon SNS) ou à une politique Amazon EC2 Auto Scaling. CloudWatch les alarmes ne déclenchent pas d'actions simplement parce qu'elles se trouvent dans un état particulier ; cet état doit avoir changé et être maintenu pendant un certain nombre de périodes. Pour plus d'informations, consultez Surveillez Amazon ECS à l'aide CloudWatch.

Pour les services dont les tâches utilisent Fargate, vous pouvez utiliser CloudWatch des alarmes permettant d'adapter et de réduire les tâches de votre service en fonction de CloudWatch des mesures, telles que l'utilisation du processeur et de la mémoire. Pour plus d'informations, consultez Mise à l'échelle automatique de votre service Amazon ECS.

Pour les clusters dont les tâches ou les services utilisent EC2, vous pouvez utiliser CloudWatch alarmes pour intégrer et redimensionner les instances de conteneurs en fonction de CloudWatch des mesures, telles que la réservation de la mémoire du cluster.

Amazon CloudWatch Journaux

Surveillez, stockez et accédez aux fichiers journaux à partir des conteneurs de vos tâches Amazon ECS en spécifiant le pilote de journal `awslogs` dans vos définitions de tâche. Pour plus d'informations, veuillez consulter Utilisation du pilote awslogs (langue française non garantie).

Vous pouvez également surveiller, stocker et accéder au système d'exploitation et aux fichiers journaux de l'agent de conteneur Amazon ECS à partir de vos instances de conteneur Amazon ECS. Cette méthode d'accès aux journaux peut être utilisée pour les conteneurs employant EC2.

Amazon CloudWatch Évènements

Mettez en correspondance les événements et transmettez-les à un ou plusieurs flux ou fonctions cibles pour effectuer des modifications, capturer des informations d'état et prendre des mesures correctives. Pour plus d'informations, voir Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge dans ce guide et EventBridge est l'évolution d'Amazon CloudWatch Évènements dans le Amazon EventBridge Guide de l'utilisateur.

AWS CloudTrail Journaux

CloudTrail fournit un enregistrement des actions entreprises par un utilisateur, un rôle ou AWS service dans Amazon ECS. Utilisation des informations collectées par CloudTrail, vous pouvez

déterminer la demande qui a été envoyée à Amazon ECS, l'adresse IP à partir de laquelle la demande a été faite, qui a effectué la demande, quand elle a été faite et des informations supplémentaires. Pour plus d'informations, consultez Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail.

AWS Trusted Advisor

Trusted Advisor s'appuie sur les meilleures pratiques apprises au service de centaines de milliers de AWS clients. Trusted Advisor inspecte votre AWS environnement, puis formule des recommandations lorsque des opportunités se présentent pour économiser de l'argent, améliorer la disponibilité et les performances du système ou contribuer à combler les failles de sécurité. Tous AWS les clients ont accès à cinq Trusted Advisor chèques. Les clients disposant d'un plan de support Business ou Enterprise peuvent tout consulter Trusted Advisor chèques.

Pour plus d'informations, consultez AWS Trusted Advisor dans le Guide de l'utilisateur Support .

Optimiseur de calcul AWS

Optimiseur de calcul AWS est un service qui analyse les paramètres de configuration et d'utilisation de votre AWS ressources. Il indique si vos ressources sont optimales et génère des recommandations d'optimisation afin de réduire les coûts et d'améliorer les performances de vos charges de travail.

Pour plus d'informations, consultez Optimiseur de calcul AWS recommandations pour Amazon ECS.

Une autre partie importante de la surveillance d'Amazon ECS consiste à surveiller manuellement les articles que CloudWatch les alarmes ne couvrent pas. Le CloudWatch, Trusted Advisor, et autres AWS les tableaux de bord de console fournissent une vue d'ensemble de l'état de votre AWS environnement. Nous vous recommandons de vérifier également les fichiers journaux de vos instances de conteneur et les conteneurs de vos tâches.

Validation de la conformité pour Amazon Elastic Container Service

Notre nouveau AWS l'expérience d'inscription n'est pas conçue pour les charges de travail réglementées. Si vous utilisez notre nouveau AWS expérience d'inscription, mais que vous souhaitez utiliser AWS pour les charges de travail régulées, vous pouvez inscrivez-vous pour AWS (avancé) ou activer les fonctionnalités avancées pour votre AWS environnement.

Pour savoir si un Service AWS s'inscrit dans le cadre de programmes de conformité spécifiques, voir Services AWS inclus dans le champ d'application du programme de conformité et choisissez le programme de conformité qui vous intéresse. Pour des informations générales, voir AWS Programmes de conformité.

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 est déterminé par la sensibilité de vos données, les objectifs de conformité de votre entreprise et les lois et réglementations applicables. Pour plus d'informations sur votre responsabilité en matière de conformité lors de l'utilisation Services AWS, voir AWS Documentation de sécurité.

Pratiques exemplaires en matière de conformité et de sécurité pour Amazon ECS

Votre responsabilité en matière de conformité lorsque vous utilisez Amazon ECS est déterminée par la sensibilité de vos données, les objectifs de conformité de votre entreprise et la réglementation en vigueur.

Normes de sécurité des données de l'industrie des cartes de paiement (PCI DSS)

Il est important que vous compreniez le flux complet des données des titulaires de cartes (CHD) au sein de l'environnement lorsque vous adhérez à la norme PCI DSS. Le flux CHD détermine l'applicabilité de la norme PCI DSS, définit les limites et les composants d'un environnement de données pour les titulaires de cartes (CDE) et, par conséquent, le champ d'application d'une évaluation PCI DSS. La détermination précise du champ d'application de la norme PCI DSS est essentielle pour définir la posture de sécurité et, en fin de compte, une évaluation réussie. Les clients doivent disposer d'une procédure de détermination du champ d'application garantissant son exhaustivité et détectant les modifications ou les écarts.

La nature temporaire des applications conteneurisées complique encore davantage l'audit des configurations. Par conséquent, les clients doivent rester informés de tous les paramètres de configuration des conteneurs afin de garantir le respect des exigences de conformité tout au long du cycle de vie d'un conteneur.

Pour plus d'informations sur la conformité à la norme PCI DSS sur Amazon ECS, consultez les livres blancs suivants.

- Architecture sur Amazon ECS pour la conformité à la norme PCI DSS

HIPAA (U.S. Loi sur la portabilité et la responsabilité en matière d'assurance maladie)

L'utilisation d'Amazon ECS avec des charges de travail qui traitent des informations de santé protégées (PHI) ne nécessite aucune configuration supplémentaire. Amazon ECS agit comme un service d'orchestration qui coordonne le lancement de conteneurs sur Amazon EC2. Il ne fonctionne pas avec ou sur les données de la charge de travail orchestrée. Conformément aux réglementations HIPAA et à l' AWS Business Associate Addendum, les PHI doivent être chiffrées en transit et au repos lorsqu'elles sont consultées par des conteneurs lancés avec Amazon ECS.

Différents mécanismes de chiffrement au repos sont disponibles avec chacun AWS option de stockage, telle qu'Amazon S3, Amazon EBS et AWS KMS. Vous pouvez déployer un réseau superposé (tel que VNS3 ou Weave Net) pour garantir le chiffrement complet des PHI transférées entre les conteneurs ou pour fournir une couche de chiffrement redondante. Vous devez également utiliser la journalisation complète et rediriger tous les journaux des conteneurs vers Amazon CloudWatch. Pour plus d'informations sur l'utilisation des pratiques exemplaires en matière de sécurité des infrastructures, consultez la section Protection des infrastructures dans le Pilier Sécurité du cadre AWS Well-Architected.

AWS Security Hub CSPM

Utiliser AWS Security Hub CSPM. Ce Service AWS fournit une vue complète de votre état de sécurité dans AWS. Security Hub CSPM utilise des contrôles de sécurité pour évaluer votre AWS ressources et pour vérifier votre conformité aux normes et aux meilleures pratiques du secteur de la sécurité. Pour obtenir la liste des services et contrôles pris en charge, voir Référence des contrôles CSPM de Security Hub.

Amazon GuardDuty avec Amazon ECS Runtime Monitoring

Amazon GuardDuty est un service de détection des menaces qui permet de protéger vos comptes, vos conteneurs, vos charges de travail et les données de votre AWS environnement. À l'aide de modèles d'apprentissage automatique (ML) et de capacités de détection des anomalies et des menaces, GuardDuty surveille en permanence les différentes sources de journaux et les activités d'exécution afin d'identifier et de hiérarchiser les risques de sécurité potentiels et les activités malveillantes dans votre environnement.

Utiliser la surveillance du temps d'exécution dans GuardDuty pour identifier les comportements malveillants ou non autorisés. Runtime Monitoring protège les charges de travail exécutées sur Fargate et EC2 grâce à une surveillance continue AWS journal et activité réseau pour identifier les comportements malveillants ou non autorisés. Runtime Monitoring utilise un outil léger et entièrement

géré GuardDuty agent de sécurité qui analyse le comportement sur l'hôte, tel que l'accès aux fichiers, l'exécution des processus et les connexions réseau. Cela couvre des problèmes tels que l'escalade des privilèges, l'utilisation d'identifiants exposés ou la communication avec des adresses IP et des domaines malveillants, ainsi que la présence de logiciels malveillants sur vos instances Amazon EC2 et vos charges de travail conteneurisées. Pour plus d'informations, voir GuardDuty Surveillance du temps d'exécution dans le GuardDuty Guide de l'utilisateur.

Recommandations relatives à la conformité

Nous vous recommandons d'impliquer rapidement les responsables des programmes de conformité au sein de votre entreprise et d'utiliser le AWS un modèle de responsabilité partagée pour identifier la propriété des contrôles de conformité afin de garantir la réussite des programmes de conformité pertinents. Pour plus d'informations, consultez AWS modèle de responsabilité partagée pour Amazon ECS.

AWS Fargate Norme fédérale de traitement de l'information (FIPS-140)

Norme fédérale de traitement de l'information (FIPS-140) est un U.S. et une norme du gouvernement canadien qui spécifie les exigences de sécurité pour les modules cryptographiques qui protègent les informations sensibles. FIPS-140 définit un ensemble de fonctions cryptographiques validées qui peuvent être utilisées pour chiffrer les données en transit et les données au repos.

Lorsque vous allumez FIPS-140 conformité, vous pouvez exécuter des charges de travail sur Fargate d'une manière conforme à FIPS-140. Pour plus d'informations sur FIPS-140 conformité, voir Norme fédérale de traitement de l'information (FIPS) 140-3.

AWS Fargate FIPS-140 Considérations

Tenez compte des points suivants lors de l'utilisation FIPS-140 conformité sur Fargate :

- FIPS-140 la conformité n'est disponible que dans AWS GovCloud (US) Régions.
- Supports Fargate FIPS-140 la version 140.3
- FIPS-140 la conformité est désactivée par défaut. Vous devez l'activer.
- Amazon CloudWatch ne prend pas en charge un point de terminaison FIPS à double pile qui peut être utilisé pour surveiller les tâches Amazon ECS dans IPv6-only configuration qui utilise FIPS-140 conformité.
- Vos tâches doivent utiliser la configuration suivante pour FIPS-140 conformité :

- Le `operatingSystemFamily` doit être `LINUX`.
- Le `cpuArchitecture` doit être `X86_64`.
- La version de plateforme Fargate doit être `1.4.0` ou ultérieure.

Utilisation de FIPS sur Fargate

Utilisez la procédure suivante pour utiliser FIPS-140 conformité sur Fargate.

1. Allumer FIPS-140 conformité. Pour plus d'informations, consultez the section called "AWS Fargate Conformité à la norme fédérale de traitement de l'information (FIPS-140)".
2. Vous pouvez éventuellement utiliser ECS Exec pour exécuter la commande suivante afin de vérifier FIPS-140 état de conformité d'un cluster.

   Remplacer *`cluster-name`* avec le nom de votre cluster, *`task-id`* avec l'ID ou l'ARN de votre tâche, et *`container-name`* avec le nom du conteneur de la tâche sur laquelle vous souhaitez exécuter la commande.

   Une valeur renvoyée de « 1 » indique que vous utilisez FIPS.

   ```
   aws ecs execute-command \
       --cluster cluster-name \
       --task task-id \
       --container container-name \
       --interactive \
       --command "cat /proc/sys/crypto/fips_enabled"
   ```

Utiliser CloudTrail pour Fargate FIPS-140 audit

CloudTrail est activé dans votre AWS compte lorsque vous créez le compte. Lorsque l'activité de l'API et de la console se produit dans Amazon ECS, cette activité est enregistrée dans un CloudTrail événement avec d'autres AWS événements de service en Historique de l'événement. Vous pouvez consulter, rechercher et télécharger les événements récents dans votre AWS compte. Pour plus d'informations, voir Afficher les événements avec CloudTrail Historique de l'événement.

Pour un enregistrement continu des événements survenus dans votre AWS compte, y compris les événements pour Amazon ECS, créez un historique qui CloudTrail permet 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 sentier enregistre les événements de toutes les régions du AWS partitionnez et remettez les fichiers journaux au compartiment Amazon S3 que vous spécifiez. En outre, vous pouvez configurer d'autres AWS services pour analyser plus en profondeur et agir en fonction des données relatives aux événements collectées dans CloudTrail journaux. Pour plus d'informations, consultez the section called "Enregistrez les appels d'API Amazon ECS à l'aide de AWS CloudTrail".

L'exemple suivant montre CloudTrail entrée de journal qui montre `PutAccountSettingDefault` Action de l'API :


```
{
    "eventVersion": "1.08",
    "userIdentity": {
        "type": "IAMUser",
        "principalId": "AIDAIV5AJI5LXF5EXAMPLE",
         "arn": "arn:aws:iam::123456789012:user/jdoe",
        "accountId": "123456789012",
        "accessKeyId": "AKIAIPWIOFC3EXAMPLE",
    },
    "eventTime": "2023-03-01T21:45:18Z",
    "eventSource": "ecs.amazonaws.com",
    "eventName": "PutAccountSettingDefault",
    "awsRegion": "us-gov-east-1",
    "sourceIPAddress": "52.94.133.131",
    "userAgent": "aws-cli/2.9.8 Python/3.9.11 Windows/10 exe/AMD64 prompt/off command/
ecs.put-account-setting",
    "requestParameters": {
        "name": "fargateFIPSMode",
        "value": "enabled"
    },
    "responseElements": {
        "setting": {
            "name": "fargateFIPSMode",
            "value": "enabled",
            "principalArn": "arn:aws:iam::123456789012:user/jdoe"
        }
    },
    "requestID": "acdc731e-e506-447c-965d-f5f75EXAMPLE",
    "eventID": "6afced68-75cd-4d44-8076-0beEXAMPLE",
    "readOnly": false,
    "eventType": "AwsApiCall",
    "managementEvent": true,
    "recipientAccountId": "123456789012",
```

```
    "eventCategory": "Management",
    "tlsDetails": {
        "tlsVersion": "TLSv1.2",
        "cipherSuite": "ECDHE-RSA-AES128-GCM-SHA256",
        "clientProvidedHostHeader": "ecs-fips.us-gov-east-1.amazonaws.com"
    }
}
```

Sécurité de l'infrastructure dans Amazon Elastic Container Service

En tant que service géré, Amazon Elastic Container Service est protégé par AWS sécurité du réseau mondial. Pour plus d'informations sur AWS services de sécurité et comment AWS protège l'infrastructure, voir AWS Sécurité du cloud. Pour concevoir votre AWS environnement utilisant les meilleures pratiques en matière de sécurité de l'infrastructure, voir Protection de l'infrastructure dans Pilier de sécurité AWS Un cadre bien conçu.

Tu utilises AWS appels d'API publiés pour accéder à Amazon ECS 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.
- Suites de chiffrement avec une confidentialité avancée parfaite (PFS) telles que DHE (Ephemeral Diffie-Hellman) ou ECDHE (courbe elliptique éphémère) Diffie-Hellman). La plupart des systèmes modernes tels que Java 7 et les versions ultérieures prennent en charge ces modes.

Vous pouvez appeler ces opérations d'API à partir de n'importe quel endroit du réseau. Amazon ECS prend en charge les stratégies d'accès basées sur les ressources, ce qui peut inclure des restrictions en fonction de l'adresse IP source. Assurez-vous donc que les politiques tiennent compte de l'adresse IP de l'emplacement du réseau. Vous pouvez également utiliser des politiques Amazon ECS pour contrôler l'accès à partir de points de terminaison Amazon Virtual Private Cloud (Amazon VPC) ou de VPC spécifiques. En fait, cela isole l'accès réseau à une ressource Amazon ECS donnée uniquement du VPC spécifique au sein du AWS réseau. Pour plus d’informations, consultez Points de terminaison VPC de l'interface Amazon ECS (AWS PrivateLink).

Points de terminaison VPC de l'interface Amazon ECS (AWS PrivateLink)

Vous pouvez améliorer le niveau de sécurité de votre VPC en configurant Amazon ECS pour utiliser un point de terminaison d’un VPC d'interface. Les points de terminaison de l'interface sont alimentés par AWS PrivateLink, une technologie qui vous permet d'accéder en privé aux API Amazon ECS

à l'aide d'adresses IP privées. AWS PrivateLink limite l'ensemble du trafic réseau entre votre VPC et Amazon ECS vers le réseau Amazon. Vous n'avez pas besoin d'une passerelle Internet, d'un périphérique NAT ni d'une passerelle privée virtuelle.

Pour plus d'informations sur AWS PrivateLink et points de terminaison VPC, voir Points de terminaison VPC dans le Guide de l'utilisateur d'Amazon VPC.

Considérations

Considérations relatives aux points de terminaison dans les régions introduites à compter du 23 décembre 2023

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Vous devez disposer des éléments suivants Region-specific Points de terminaison VPC :

  > **Note**
  >
  > Si vous ne configurez pas tous les points de terminaison, votre trafic passera par les points de terminaison publics, et non par votre point de terminaison de VPC.

  - `com.amazonaws.region.ecs-agent`
  - `com.amazonaws.region.ecs-telemetry`
  - `com.amazonaws.region.ecs`

  Par exemple, la région Canada Ouest (Calgary) (ca-west-1) a besoin des points de terminaison de VPC suivants :

  - `com.amazonaws.ca-west-1.ecs-agent`
  - `com.amazonaws.ca-west-1.ecs-telemetry`
  - `com.amazonaws.ca-west-1.ecs`
- Lorsque vous utilisez un modèle pour créer AWS ressources de la nouvelle région et le modèle a été copié à partir d'une région introduite avant le 23 décembre 2023, en fonction de la région de copie, effectuez l'une des opérations suivantes.

  Par exemple, la région source de la copie est USA Est (Virginie du Nord) (us-east-1). La région destination de la copie est Canada Ouest (Calgary) (ca-west-1)

Configuration	Action
La région source copiée ne possède aucun point de terminaison VPC.	Créez les trois points de terminaison de VPC pour la nouvelle région (par exemple, `com.amazonaws.ca-west-1.ecs-agent`).
La région copiée contient Region-specific Points de terminaison VPC.	a. Créez les trois points de terminaison de VPC pour la nouvelle région (par exemple, `com.amazonaws.ca-west-1.ecs-agent`). b. Supprimez les trois points de terminaison VPC pour la région source (par exemple, `com.amazonaws.us-east-1.ecs-agent`).

Éléments à prendre en compte pour les points de terminaison de VPC Amazon ECS pour Fargate

Lorsqu'il existe un point de terminaison de VPC pour `ecr.dkr` et `ecr.api` dans le même VPC dans lequel une tâche Fargate est déployée, le point de terminaison de VPC est utilisé. S'il n'y a pas de point de terminaison de VPC, l'interface Fargate est utlilisée.

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Les tâches utilisant Fargate ne nécessitent pas les points de terminaison VPC d'interface pour Amazon ECS, mais vous pourriez avoir besoin de points de terminaison VPC d'interface pour Amazon ECR, Secrets Manager ou Amazon CloudWatch Journaux décrits dans les points suivants.
  - Pour autoriser vos tâches à extraire des images privées depuis Amazon ECR, vous devez créer les points de terminaison d'un VPC d'interface pour Amazon ECR. Pour de plus amples

informations, veuillez consulter Points de terminaison de VPC d'interface (AWS PrivateLink) dans le Guide de l'utilisateur Amazon Elastic Container Registry.

> **Important**
>
> - Si vous configurez Amazon ECR pour utiliser un point de terminaison de VPC d'interface, vous pouvez créer un rôle d'exécution de tâche incluant des clés de condition pour restreindre l'accès à un VPC ou à un point de terminaison de VPC spécifique. Pour plus d'informations, consultez Tâches Fargate extrayant des images Amazon ECR via les autorisations des points de terminaison d'interface.
> - Si vos tâches se situent dans un IPv6-only configurez et utilisez un URI d'image Amazon ECR dualstack, notez qu'Amazon ECR ne prend pas en charge les points de terminaison VPC à interface dualstack. Pour plus d'informations, consultez la section Premiers pas avec les requêtes IPv6 dans le Guide de l'utilisateur Amazon Elastic Container Registry.

- Pour autoriser vos tâches à extraire des données sensibles depuis Secrets Manager, vous devez créer les points de terminaison d'un VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
- Si votre VPC ne possède pas de passerelle Internet et que vos tâches utilisent `awslogs` pilote de journal auquel envoyer les informations de journal CloudWatch Journaux, vous devez créer une interface pour laquelle vous devez créer un point de terminaison VPC CloudWatch Journaux. Pour plus d'informations, voir En utilisant CloudWatch Journaux avec les points de terminaison VPC d'interface dans le Amazon CloudWatch Guide de l'utilisateur des journaux.

- Les points de terminaison d'un VPC ne prennent pas en charge les demandes inter-régions pour le moment. Veillez à créer votre point de terminaison dans la même région que celle dans laquelle vous souhaitez envoyer vos appels d'API à Amazon ECS. Supposons, par exemple, que vous souhaitiez exécuter des tâches dans USA Est (Virginie du Nord). Vous devez ensuite créer le point de terminaison Amazon ECS VPC dans USA Est (Virginie du Nord). Un point de terminaison d'un VPC Amazon ECS créé dans une autre région ne peut pas exécuter de tâche dans USA Est (Virginie du Nord).
- Prise en charge des terminaux VPC uniquement Amazon-provided DNS via Amazon Route 53. Si vous souhaitez utiliser votre propre DNS, vous pouvez utiliser le transfert DNS conditionnel. Pour en savoir plus, consultez Jeux d'options DHCP dans le Guide de l'utilisateur Amazon VPC.

- Le groupe de sécurité attaché au point de terminaison d'un VPC doit autoriser les connexions entrantes sur le port TCP 443 à partir du sous-réseau privé du VPC.
- La gestion Service Connect du proxy Envoy utilise le point de terminaison d'un VPC `com.amazonaws.region.ecs-agent`. Lorsque vous n'utilisez pas les points de terminaison d'un VPC, la gestion Service Connect du proxy Envoy utilise le point de terminaison `ecs-sc` de cette région. Pour obtenir la liste des points de terminaison Amazon ECS dans chaque région, veuillez consulter Points de terminaison et quotas Amazon ECS (langue française non garantie).

Considérations relatives aux points de terminaison de VPC Amazon ECS

Avant de configurer des points de terminaison d'un VPC d'interface pour Amazon ECS, tenez compte de ce qui suit :

- Les tâches qui utilisent EC2 nécessitent que les instances de conteneur sur lesquelles elles sont lancées exécutent la version `1.25.1` ou ultérieure de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Gestion des instances de conteneur Amazon ECS Linux.
- Pour autoriser vos tâches à extraire des données sensibles depuis Secrets Manager, vous devez créer les points de terminaison d'un VPC d'interface pour Secrets Manager. Pour de plus amples informations, veuillez consulter Utilisation de Secrets Manager avec des points de terminaison de VPC dans le Guide de l'utilisateur AWS Secrets Manager .
- Si votre VPC ne possède pas de passerelle Internet et que vos tâches utilisent `awslogs` pilote de journal auquel envoyer les informations de journal CloudWatch Journaux, vous devez créer une interface pour laquelle vous devez créer un point de terminaison VPC CloudWatch Journaux. Pour plus d'informations, voir En utilisant CloudWatch Journaux avec les points de terminaison VPC d'interface dans le Amazon CloudWatch Guide de l'utilisateur des journaux.
- Les points de terminaison d'un VPC ne prennent pas en charge les demandes inter-régions pour le moment. Veillez à créer votre point de terminaison dans la même région que celle dans laquelle vous souhaitez envoyer vos appels d'API à Amazon ECS. Supposons, par exemple, que vous souhaitiez exécuter des tâches dans USA Est (Virginie du Nord). Vous devez ensuite créer le point de terminaison Amazon ECS VPC dans USA Est (Virginie du Nord). Un point de terminaison de VPC Amazon ECS créé dans une autre région ne peut pas exécuter de tâche dans la région USA Est (Virginie du Nord).
- Prise en charge des terminaux VPC uniquement Amazon-provided DNS via Amazon Route 53. Si vous souhaitez utiliser votre propre DNS, vous pouvez utiliser le transfert DNS conditionnel. Pour en savoir plus, consultez Jeux d'options DHCP dans le Guide de l'utilisateur Amazon VPC.

- Le groupe de sécurité attaché au point de terminaison d'un VPC doit autoriser les connexions entrantes sur le port TCP 443 à partir du sous-réseau privé du VPC.

Compréhension des modèles de dénomination des points de terminaison Amazon ECS

L'agent de conteneur Amazon ECS utilise DiscoverPollEndpoint API permettant de trouver l'agent et les points de terminaison de télémétrie auxquels se connecter. Les noms d'hôte renvoyés varient selon que l'appel utilise ou non `com.amazonaws.region.ecs` point de terminaison VPC d'interface.

- Lorsque l'appel passe par `ecs` interface point de terminaison VPC, les points de terminaison renvoyés utilisent les noms DNS privés de base `ecs-a.region.amazonaws.com` et `ecs-t.region.amazonaws.com`. Un identifiant numérique apparaît dans le chemin de l'URL, par exemple `https://ecs-a.region.amazonaws.com/acs/1/`.
- Lorsque l'appel ne passe pas par le `ecs` interface point de terminaison VPC, les points de terminaison renvoyés peuvent utiliser des noms d'hôte numérotés, tels que `ecs-a-1.region.amazonaws.com` et `ecs-t-1.region.amazonaws.com`. Ces noms d'hôte se réfèrent à des points de terminaison publics.

Il n'est pas nécessaire de créer des enregistrements DNS pour les noms d'hôtes numérotés. Si vous utilisez vos propres zones hébergées privées Route 53 au lieu du DNS privé du terminal, créez des zones hébergées uniquement pour les domaines suivants :

- `ecs.region.amazonaws.com`
- `ecs-a.region.amazonaws.com`
- `ecs-t.region.amazonaws.com`

Résolution des problèmes de connectivité des agents via les terminaux VPC

Si vos instances de conteneur se connectent à des noms d'hôte numérotés tels que `ecs-a-1.region.amazonaws.com`, l'agent DiscoverPollEndpoint l'appel ne passe pas par le `ecs` point de terminaison VPC d'interface. Vérifiez les points suivants à partir de l'instance de conteneur :

- `ecs.region.amazonaws.com` se résout aux adresses IP privées de votre `ecs` point final.

- Si l'agent utilise un proxy HTTP, soit le proxy résout `ecs.region.amazonaws.com` à votre `ecs` point de terminaison, ou le nom d'hôte se trouve dans `NO_PROXY`. Sinon, le proxy peut envoyer l'appel via le chemin public même si le DNS de l'instance est correct. Pour plus d'informations, consultez Utilisation d'un proxy HTTP pour les instances de conteneur Amazon ECS Linux.

Une fois l'appel passé par `ecs` point de terminaison auquel l'agent se connecte `ecs-a.region.amazonaws.com` et `ecs-t.region.amazonaws.com`. Assurez-vous que vous avez également créé le `ecs-agent` et `ecs-telemetry` interfacer les points de terminaison VPC et que ces noms d'hôtes soient résolus en adresses IP privées. Pour plus d'informations, consultez Création de points de terminaison de VPC pour Amazon ECS.

Création de points de terminaison de VPC pour Amazon ECS

Pour créer le point de terminaison VPC pour le service Amazon ECS, utilisez le Accédez à un AWS service utilisant une interface VPC endpoint procédure dans le Guide de l'utilisateur d'Amazon VPC pour créer les points de terminaison suivants. Si vous disposez d'instances de conteneur dans votre VPC, vous devriez créer les points de terminaison dans l'ordre dans lequel ils sont répertoriés. Si vous envisagez de créer vos instances de conteneur après la création de votre point de terminaison d'un VPC, alors l'ordre n'est pas important.

> **Note**
>
> Si vous ne configurez pas tous les points de terminaison, votre trafic passera par les points de terminaison publics, et non par votre point de terminaison de VPC.
> Lorsque vous créez des points de terminaison, Amazon ECS crée également un nom DNS privé pour le point de terminaison. Par exemple, `ecs-a.region.amazonaws.com` pour ecs-agent et `ecs-t.region.amazonaws.com` pour ecs-telemetry.

- `com.amazonaws.region.ecs-agent`
- `com.amazonaws.region.ecs-telemetry`
- `com.amazonaws.region.ecs`

> **Note**
>
> *region* représente l'identifiant de région pour un AWS Région prise en charge par Amazon ECS, telle que `us-east-2` pour la région de l'Est des États-Unis (Ohio).

Le point de terminaison `ecs-agent` utilise l'API `ecs:poll`, et le point de terminaison `ecs-telemetry` utilise l'API `ecs:poll` et `ecs:StartTelemetrySession`.

Si vous avez des tâches existantes qui utilisent le type de lancement EC2, une fois que vous avez créé les points de terminaison d'un VPC, chaque instance de conteneur a besoin de récupérer la nouvelle configuration. Pour ce faire, vous devez redémarrer chaque instance de conteneur ou redémarrer l'agent de conteneur Amazon ECS sur chaque instance de conteneur. Faites ce qui suit pour redémarrer l'agent de conteneur.

Pour redémarrer l'agent de conteneur Amazon ECS

1. Connectez-vous à votre instance de conteneur via SSH.
2. Arrêtez l'agent de conteneur .

```
sudo docker stop ecs-agent
```

3. Démarrer l'agent de conteneur.

```
sudo docker start ecs-agent
```

Une fois les points de terminaison d'un VPC créés et l'agent de conteneur Amazon ECS redémarré sur chaque instance de conteneur, toutes les tâches nouvellement lancées adopteront la nouvelle configuration.

Création d'une stratégie de point de terminaison d'un VPC pour Amazon ECS

Vous pouvez attacher une stratégie de point de terminaison au point de terminaison d'un VPC qui contrôle l'accès à Amazon ECS. La politique spécifie les informations suivantes :

- Le principal qui peut exécuter des actions.
- Les actions qui peuvent être effectuées.
- Les ressources sur lesquelles les actions peuvent être exécutées.

Pour plus d'informations, consultez Contrôle de l'accès aux services avec points de terminaison d'un VPC dans le Guide de l'utilisateur Amazon VPC.

Exemple : stratégie de point de terminaison d’un VPC pour les actions de l'API Amazon ECS

Voici un exemple de politique de point de terminaison pour Amazon ECS. Lorsqu'elle est attachée à un point de terminaison, cette politique accorde l'accès à l'autorisation de créer et de répertorier des clusters. Les actions `CreateCluster` et `ListClusters` n'acceptent aucune ressource. La définition de ressource est donc définie sur * pour toutes les ressources.

```
{
   "Statement":[
    {
      "Principal":"*",
      "Effect": "Allow",
      "Action": [
        "ecs:CreateCluster",
        "ecs:ListClusters"
      ],
      "Resource": [
        "*"
      ]
    }
  ]
}
```

AWS modèle de responsabilité partagée pour Amazon ECS

La sécurité et la conformité sont une responsabilité partagée entre AWS et le client. Ce modèle partagé peut contribuer à alléger la charge opérationnelle du client car AWS exploite, gère et contrôle les composants depuis le système d'exploitation hôte et la couche de virtualisation jusqu'à la sécurité physique des installations dans lesquelles le service fonctionne. Le client assume toujours la responsabilité et la gestion du système d’exploitation « invité » (notamment les mises à jour et les correctifs de sécurité), d’autres éléments applicatifs connexes, de même que la configuration du pare-feu du groupe de sécurité fourni par AWS . Les clients doivent choisir avec soin les services, car leurs responsabilités varient en fonction des services utilisés, de l’intégration de ces services dans leur environnement informatique, ainsi que les cadres législatifs et règlementaires applicables. La nature de cette responsabilité partagée assure également la flexibilité et le contrôle client qui permettent le déploiement.

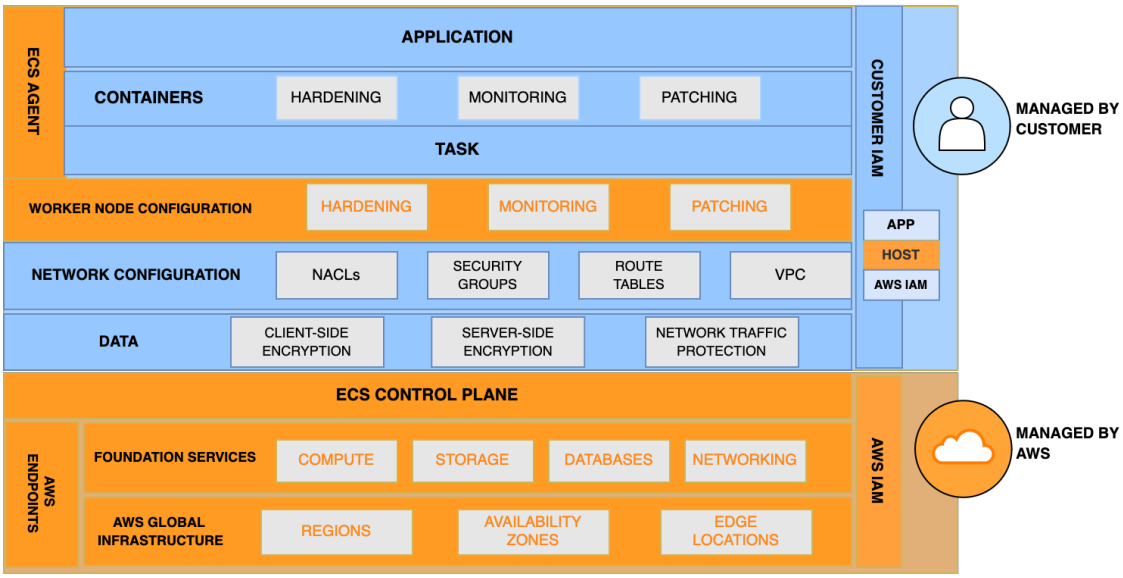
Fargate

L'illustration suivante montre le modèle de responsabilité partagée pour le type de lancement Fargate. Fargate exécute chaque charge de travail dans un environnement matériel virtualisé isolé. Par conséquent, chaque tâche bénéficie d'une capacité d'infrastructure dédiée. Les charges de travail conteneurisées exécutées sur Fargate ne partagent pas de système d'exploitation, de noyau Linux, d'interface réseau, de stockage éphémère, d'UC ou de mémoire avec d'autres tâches. Lorsqu'ils utilisent Fargate, les clients ne sont pas responsables de la sécurisation de l'infrastructure informatique qui gère leurs conteneurs. Fargate fournira et corrigera l'infrastructure sur laquelle s'exécutent les charges de travail des clients. Pour plus d'informations, consultez Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS.

Vous êtes responsable de la gestion des ressources suivantes :

- Configuration réseau, y compris VPC, NACL, groupes de sécurité et tables de routage
- Chiffrement du stockage des clients et des services. Pour plus d'informations, consultez Options de stockage pour les tâches Amazon ECS.
- Images de conteneur. Pour plus d'informations, consultez Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS.
- Autorisations IAM pour les applications à l'aide du rôle de tâche. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.

AWS Shared Responsibility Model for Amazon ECS with Fargate



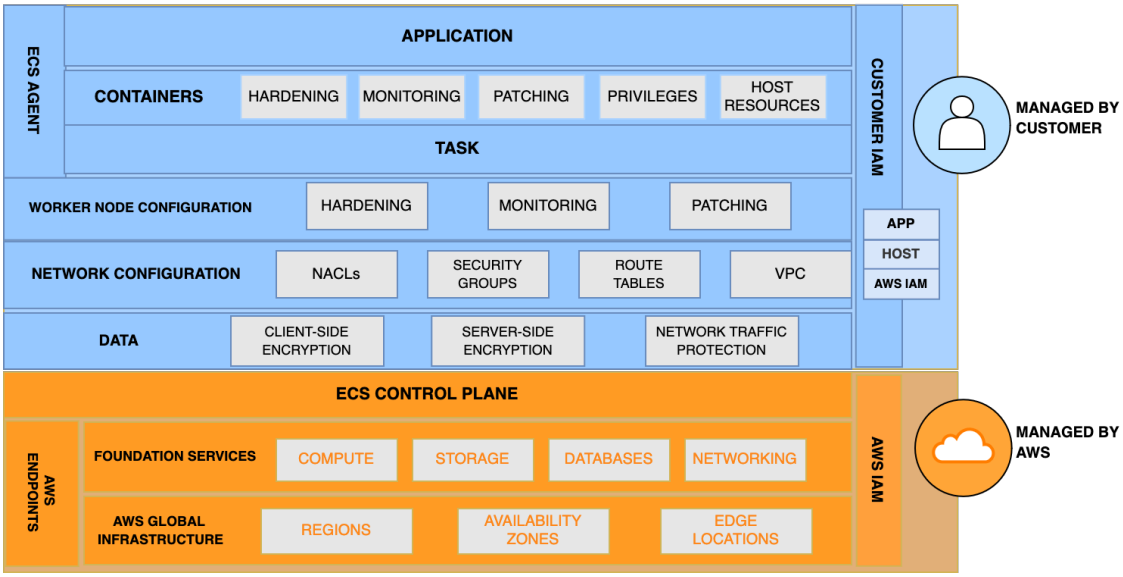
EC2

L’illustration suivante montre le modèle de responsabilité partagée pour EC2. Lorsque vous exécutez des tâches sur des instances EC2, vous êtes responsable de la maintenance de vos instances EC2 en plus des ressources suivantes :

- L’agent Amazon ECS.
- L'AMI de l'instance EC2, y compris les correctifs et le renforcement.
- Configuration réseau, y compris VPC, NACL, groupes de sécurité et tables de routage.
- Chiffrement du stockage des clients et des services. Pour plus d’informations, consultez Options de stockage pour les tâches Amazon ECS.
- Images de conteneur. Pour plus d’informations, consultez Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS.
- Autorisations IAM pour les applications à l’aide du rôle de tâche. Pour plus d’informations, consultez rôle IAM de tâche Amazon ECS.
- Configuration des instances de conteneur pour bloquer l'accès des conteneurs au service de métadonnées d'instance Amazon EC2 (IMDS). Pour plus d’informations, consultez Recommandations relatives aux rôles.
- Considérations relatives à la sécurité pour les charges de travail colocalisées. Contrairement à Fargate, il n'existe aucune isolation des tâches sur les instances de conteneur EC2. Les conteneurs peuvent potentiellement accéder aux informations d'identification, aux variables d'environnement et aux fichiers temporaires provenant d'autres tâches de la même instance, y compris les données laissées par les tâches exécutées précédemment. Pour les applications sensibles, configurez votre environnement de manière à ce que chaque application s'exécute sur des instances dédiées.

Le schéma suivant illustre le modèle de responsabilité partagée pour EC2, indiquant quelles responsabilités de sécurité sont gérées par AWS et qui sont les vôtres.

AWS Shared Responsibility Model for Amazon ECS



Modèle de responsabilité partagée pour les instances gérées Amazon ECS

Amazon ECS Managed Instances fournit une solution gérée pour les charges de travail conteneurisées qui associe la simplicité opérationnelle de Fargate à l'accès à la gamme complète de types et de fonctionnalités d'instances Amazon EC2. AWS gère le provisionnement, l'application de correctifs, la mise à l'échelle et la maintenance de l'infrastructure, tandis que les clients conservent le contrôle de leurs applications et de leurs configurations spécifiques.

Contrairement à Fargate, les charges de travail conteneurisées exécutées sur les instances gérées Amazon ECS partagent le système d'exploitation, le noyau Linux, l'interface réseau, le stockage éphémère, l'UC, la mémoire et les ressources du GPU avec les autres tâches de la même instance. Amazon ECS optimise l'utilisation de l'infrastructure en confiant plusieurs tâches à des instances plus importantes afin de minimiser la capacité inutilisée.

AWS responsabilités

Lorsque vous utilisez des instances gérées Amazon ECS, AWS est responsable de :

- allocation des instances et gestion du cycle de vie ;
- application de correctifs au système d'exploitation et mises à jour de sécurité ;
- mise à l'échelle et l'optimisation de l'infrastructure ;

- remplacement et maintenance de l'instance (durée de vie maximale de l'instance de 21 jours) ;
- restrictions de contrôle d'accès (pas d'accès SSH, pas d'accès au gestionnaire de sessions SSM) ;
- chiffrement du stockage d'instance Amazon EC2, c'est-à-dire le stockage directement attaché à l'instance. Pour de plus amples informations, consultez la section Protection des données dans Amazon EC2.
- Amazon ECS gère les volumes attachés aux instances Amazon EC2 au moment de leur création, y compris les volumes racine et les volumes de données.
- Amazon ECS utilise des instances gérées par Amazon EC2 en arrière-plan. Pour plus d'informations sur les instances gérées par Amazon EC2, consultez la section Sécurité dans Amazon EC2.

Responsabilités client

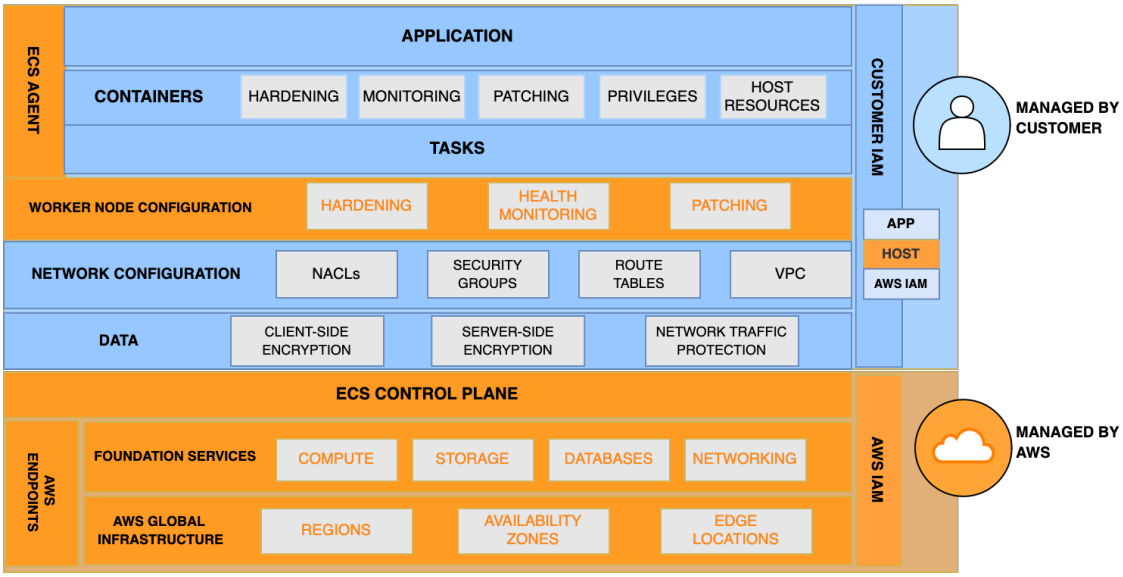
Vous êtes responsable de la gestion des ressources suivantes :

- Configuration réseau, y compris VPC, NACL, groupes de sécurité et tables de routage
- Chiffrement du stockage des clients et des services. Pour plus d'informations, consultez Options de stockage pour les tâches Amazon ECS.
- Images de conteneur. Pour plus d'informations, consultez Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS.
- Autorisations IAM pour les applications à l'aide du rôle de tâche. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.
- Application-level configuration et surveillance
- Surveillance de la sécurité de vos conteneurs
- Définitions des tâches et des services
- Considérations relatives à la sécurité pour les charges de travail colocalisées. Contrairement à Fargate, il n'existe aucune isolation des tâches sur les instances gérées Amazon ECS. Les conteneurs peuvent potentiellement accéder aux informations d'identification, aux variables d'environnement et aux fichiers temporaires provenant d'autres tâches de la même instance, y compris les données laissées par les tâches exécutées précédemment.
- Configurations de conteneurs privilégiées et fonctionnalités Linux améliorées (CAP_NET_ADMIN, CAP_BPF, CAP_PERFMON, etc.) lorsqu'elles sont activées
- Opérations de gestion via l'API Amazon ECS (l'accès direct aux instances via SSH ou SSM n'est pas disponible)

Pour plus d'informations sur AWS et les responsabilités des clients en matière de sécurité des instances gérées Amazon ECS, voir Considérations de sécurité pour les instances gérées Amazon ECS.

Le schéma suivant illustre le modèle de responsabilité partagée pour les instances gérées Amazon ECS, indiquant quelles responsabilités de sécurité sont gérées par AWS et qui sont les vôtres.

AWS Shared Responsibility Model for Amazon ECS with Managed Instances



Pratiques exemplaires en matière de sécurité réseau pour Amazon ECS

La sécurité des réseaux est un vaste sujet qui englobe plusieurs sous-thèmes. Il s'agit notamment du chiffrement en transit, de la segmentation et de l'isolation du réseau, du pare-feu, du routage du trafic et de l'observabilité.

Chiffrement en transit

Le chiffrement du trafic réseau empêche les utilisateurs non autorisés d'intercepter et de lire des données lorsque celles-ci sont transmises sur un réseau. Avec Amazon ECS, le chiffrement réseau peut être mis en œuvre de l'une des façons suivantes.

- À l'aide des instances Nitro :

  Par défaut, le trafic est automatiquement chiffré entre les types d'instances Nitro suivants : C5n, G4, I3en, M5dn, M5n, P3dn, R5dn et R5n. Le trafic n'est pas chiffré lorsqu'il est acheminé via une passerelle de transit, un équilibreur de charge ou un intermédiaire similaire.

  - Chiffrement en transit
  - Annonce des nouveautés de 2019
  - Discussion lors de re:Inforce 2019
- À l'aide de Server Name Indication (SNI) et d'un Application Load Balancer :

  L'Application Load Balancer (ALB) et le Network Load Balancer (NLB) prennent en charge Server Name Indication (SNI). En utilisant SNI, vous pouvez placer plusieurs applications sécurisées derrière un seul écouteur. Pour cela, chaque écouteur possède son propre certificat TLS. Nous vous recommandons de fournir des certificats pour l'équilibreur de charge à l'aide d' AWS Certificate Manager (ACM), puis de les ajouter à la liste des certificats de l'écouteur. Le AWS L'équilibreur de charge utilise un algorithme de sélection de certificat intelligent avec SNI. Si le nom d'hôte fourni par un client correspond à un seul certificat de la liste de certificats, l'équilibreur de charge sélectionne ce certificat. Si un nom d'hôte fourni par un client correspond à plusieurs certificats de la liste, l'équilibreur de charge sélectionne un certificat pouvant être pris en charge par le client. Les exemples incluent un certificat auto-signé ou un certificat généré via l'ACM.

  - SNI avec un Application Load Balancer
  - SNI avec un Network Load Balancer
- End-to-end chiffrement à l'aide de certificats TLS :

  Cela implique le déploiement d'un certificat TLS avec la tâche. Il peut s'agir d'un certificat auto-signé ou d'un certificat délivré par une autorité de certification fiable. Vous pouvez obtenir le certificat en indiquant un secret pour le certificat. Sinon, vous pouvez choisir d’exécuter un conteneur qui envoie une demande de signature de certificat (CSR, Certificate Signing Request) à ACM, puis monte le secret obtenu sur un volume partagé.

  - Maintaining transport layer security all the way to your containers using the Network Load Balancer with Amazon ECS part 1
  - Maintien de la sécurité de la couche de transport (TLS) jusqu'à votre conteneur, partie 2 : Utilisation Autorité de certification privée AWS

Mise en réseau des tâches

Les recommandations suivantes tiennent compte du fonctionnement d'Amazon ECS. Amazon ECS n'utilise pas de réseau superposé. Au lieu de cela, les tâches sont configurées pour fonctionner dans différents modes réseau. Par exemple, les tâches configurées pour utiliser le mode `bridge` acquièrent une adresse IP non routable auprès d'un réseau Docker exécuté sur chaque hôte. Les

tâches configurées pour utiliser le mode réseau `awsvpc` acquièrent une adresse IP auprès du sous-réseau de l'hôte. Les tâches configurées avec la mise en réseau `host` utilisent l'interface réseau de l'hôte. `awsvpc` est le mode réseau préféré. C'est parce qu'il s'agit du seul mode que vous pouvez utiliser pour attribuer des groupes de sécurité aux tâches. C'est également le seul mode disponible pour AWS Fargate tâches sur Amazon ECS.

Groupes de sécurité pour les tâches

Nous vous recommandons de configurer vos tâches de manière à utiliser le mode réseau `awsvpc`. Une fois que vous avez configuré votre tâche pour utiliser ce mode, l'agent Amazon ECS provisionne et attache automatiquement une Interface réseau Elastic (ENI) à la tâche. Lorsque l'ENI est provisionné, la tâche est inscrite dans un AWS groupe de sécurité. Le groupe de sécurité agit en tant que pare-feu virtuel que vous pouvez utiliser afin de contrôler le trafic entrant et sortant.

Si vous utilisez un pare-feu personnalisé avec des tâches ou des services, ajoutez une règle sortante pour autoriser le trafic vers les points de terminaison de gestion des agents Amazon ECS (« `ecs-a-*.`*`region`*`.amazonaws.com` »), les points de terminaison de télémétrie (« `ecs-t-*.`*`region`*`.amazonaws.com` ») et les points de terminaison de gestion Service Connect Envoy (« `ecs-sc.`*`region`*`.api.aws`»).

Si vous utilisez des instances gérées Amazon ECS, ajoutez également une règle sortante pour autoriser le trafic vers les points de terminaison du plan de données (»`ecs-d-*.`*`region`*`.api.aws`«). Vos instances gérées utilisent ces points de terminaison pour la livraison des journaux système et d'autres opérations sur le plan de données.

AWS PrivateLink et Amazon ECS

AWS PrivateLink est une technologie réseau qui vous permet de créer des points de terminaison privés pour différents AWS services, notamment Amazon ECS. Les points de terminaison sont nécessaires dans les environnements de test (sandbox) dans lesquels aucune passerelle Internet (IGW) n'est connectée à Amazon VPC et aucune alternative ne mène à Internet. En utilisant AWS PrivateLink garantit que les appels vers le service Amazon ECS restent dans l'Amazon VPC et ne passent pas par Internet. Pour obtenir des instructions sur la création AWS PrivateLink points de terminaison pour Amazon ECS et autres services associés, voir Interface Amazon ECS, points de terminaison Amazon VPC.

> **Important**
>
> AWS Fargate les tâches ne nécessitent pas de AWS PrivateLink point de terminaison pour Amazon ECS.

Amazon ECR et Amazon ECS prennent tous deux en charge les stratégies de point de terminaison. Ces stratégies vous permettent d'affiner l'accès aux API d'un service. Par exemple, vous pouvez créer une stratégie de point de terminaison pour Amazon ECR qui autorise uniquement le transfert d'images vers des registres de comptes spécifiques AWS . Une telle stratégie pourrait être utilisée pour empêcher l'exfiltration de données par le biais d'images de conteneurs tout en permettant aux utilisateurs de les envoyer vers des registres Amazon ECR autorisés. Pour plus d'informations, veuillez consulter Utilisation des stratégies de point de terminaison pour contrôler l'accès à des points de terminaison d'un VPC.

La politique suivante permet à tous AWS les principaux de votre compte pour effectuer toutes les actions uniquement sur vos référentiels Amazon ECR :

```
{
  "Statement": [
    {
      "Sid": "LimitECRAccess",
      "Principal": "*",
      "Action": "*",
      "Effect": "Allow",
      "Resource": "arn:aws:ecr:region:account_id:repository/*"
    }
  ]
}
```

Vous pouvez encore améliorer cela en définissant une condition qui utilise la nouvelle propriété `PrincipalOrgID`. Cela empêche l'envoi et l'extraction d'images par un responsable IAM qui ne fait pas partie de votre AWS Organizations. Pour plus d'informations, voir aws :PrincipalOrgIDENTIFIANT.

Nous avons recommandé d'appliquer la même politique aux points de terminaison `com.amazonaws.region.ecr.dkr` et `com.amazonaws.region.ecr.api`.

Paramètres d'agent de conteneur

Le fichier de configuration de l'agent de conteneur Amazon ECS inclut plusieurs variables d'environnement liées à la sécurité du réseau. `ECS_AWSVPC_BLOCK_IMDS` et `ECS_ENABLE_TASK_IAM_ROLE_NETWORK_HOST` sont utilisées pour bloquer l'accès d'une tâche aux métadonnées Amazon EC2. `HTTP_PROXY` est utilisée pour configurer l'agent pour qu'il passe par un proxy HTTP afin de se connecter à Internet. Pour obtenir des instructions sur la configuration de l'agent et de l'environnement d'exécution Docker pour le routage via un proxy, veuillez consulter Configuration du proxy HTTP (langue française non garantie).

> **Important**
>
> Ces paramètres ne sont pas disponibles lorsque vous utilisez AWS Fargate.

Recommandations de sécurité réseau

Nous vous recommandons de procéder comme suit lors de la configuration de votre Amazon VPC, de vos équilibreurs de charge et de votre réseau.

Utilisation du chiffrement réseau le cas échéant avec Amazon ECS

Vous devez utiliser le chiffrement réseau, le cas échéant. Certains programmes de conformité, tels que la norme PCI DSS, exigent que vous chiffriez les données en transit si celles-ci contiennent des données relatives au titulaire de la carte. Si votre charge de travail présente des exigences similaires, configurez le chiffrement réseau.

Les navigateurs modernes avertissent les utilisateurs lorsqu'ils se connectent à des sites non sécurisés. Si votre service est géré par un équilibreur de charge public, utilisez TLS/SSL pour chiffrer le trafic du navigateur du client vers l'équilibreur de charge et le rechiffrer vers le backend si nécessaire.

Utiliser **`awsvpc`** mode réseau et groupes de sécurité pour contrôler le trafic entre les tâches et les autres ressources dans Amazon ECS

Vous devez utiliser le mode réseau `awsvpc` et les groupes de sécurité lorsque vous devez contrôler le trafic entre les tâches et entre les tâches et les autres ressources du réseau. Si votre service se trouve derrière un ALB, utilisez des groupes de sécurité pour autoriser uniquement le trafic entrant

provenant d'autres ressources réseau utilisant le même groupe de sécurité que votre ALB. Si votre application se trouve derrière un NLB, configurez le groupe de sécurité de la tâche pour autoriser uniquement le trafic entrant provenant de la plage d'adresses CIDR Amazon VPC et des adresses IP statiques attribuées au NLB.

Les groupes de sécurité doivent également être utilisés pour contrôler le trafic entre les tâches et les autres ressources au sein d'Amazon VPC, telles que les bases de données Amazon RDS.

Création de clusters Amazon ECS dans des VPC Amazon distincts lorsque le trafic réseau doit être strictement isolé

Vous devez créer des clusters dans des Amazon VPC distincts lorsque le trafic réseau doit être strictement isolé. Évitez d'exécuter des charges de travail soumises à des exigences de sécurité strictes sur des clusters dont les charges de travail ne sont pas tenues de respecter ces exigences. Lorsqu'une isolation réseau stricte est obligatoire, créez des clusters dans des Amazon VPC distincts et exposez les services de manière sélective à d'autres Amazon VPC à l'aide des points de terminaison Amazon VPC. Pour plus d'informations, consultez Points de terminaison d'un VPC.

Configurer AWS PrivateLink points de terminaison lorsque cela est justifié pour Amazon ECS

Vous devez configurer AWS PrivateLink points de terminaison lorsque cela est justifié. Si votre politique de sécurité vous empêche de connecter une passerelle Internet (IGW) à vos Amazon VPC, configurez AWS PrivateLink points de terminaison pour Amazon ECS et d'autres services tels qu'Amazon ECR, AWS Secrets Manager, et Amazon CloudWatch.

Utilisation des journaux de flux Amazon VPC pour analyser le trafic à destination et en provenance de tâches de longue durée dans Amazon ECS

Vous devez utiliser les journaux de flux Amazon VPC pour analyser le trafic à destination et en provenance de tâches de longue durée. Les tâches qui utilisent le mode réseau `awsvpc` obtiennent leur propre ENI. Ainsi, vous pouvez surveiller le trafic à destination et en provenance de tâches individuelles à l'aide des journaux de flux Amazon VPC. Une récente mise à jour des journaux de flux Amazon VPC (v3) enrichit les journaux avec des métadonnées de trafic, notamment l'ID du VPC, l'ID de sous-réseau et l'ID d'instance. Ces métadonnées peuvent être utilisées pour affiner une enquête. Pour plus d'informations, veuillez consulter Journaux de flux Amazon VPC.

> **Note**
>
> En raison de la nature temporaire des conteneurs, les journaux de flux ne sont pas toujours un moyen efficace d'analyser les modèles de trafic entre les différents conteneurs ou entre les conteneurs et les autres ressources du réseau.

Pratiques exemplaires en matière de sécurité des tâches et des conteneurs Amazon ECS

Vous devez considérer l'image du conteneur comme la première ligne de défense contre une attaque. Une image peu sécurisée et mal construite peut permettre à un attaquant d'échapper aux limites du conteneur et d'accéder à l'hôte. Vous devez procéder comme suit pour atténuer le risque que cela se produise.

Nous vous recommandons d'effectuer les opérations suivantes lorsque vous configurez vos tâches et conteneurs.

Créez un minimum d'images ou utilisez des images sans distribution

Commencez par retirer tous les fichiers binaires superflus de l'image de conteneur. Si vous utilisez une image inconnue provenant de la Galerie publique Amazon ECR, inspectez-la pour faire référence au contenu de chacune des couches du conteneur. Pour ce faire, vous pouvez utiliser une application telle que Dive.

Vous pouvez également utiliser des images sans distribution qui incluent uniquement votre application et ses dépendances d'exécution. Elles ne contiennent pas de gestionnaires de packages ou de shells. Les images sans distribution améliorent le « rapport signal/bruit émis par les scanners et réduisent la charge liée à l'établissement de la provenance à ce dont vous avez besoin ». Pour plus d'informations, consultez le GitHub documentation sur affolé.

Docker dispose d'un mécanisme permettant de créer des images à partir d'une image minimale réservée connue sous le nom de scratch. Pour plus d'informations, veuillez consulter Création d'une image parent simple à l'aide de scratch dans la documentation Docker (langue française non garantie). Avec des langages tels que Go, vous pouvez créer un fichier binaire lié statique et le référencer dans votre Dockerfile. L'exemple suivant montre comment procéder.

```
############################
```

```
# STEP 1 build executable binary
############################
FROM golang:alpine AS builder
# Install git.
# Git is required for fetching the dependencies.
RUN apk update && apk add --no-cache git
WORKDIR $GOPATH/src/mypackage/myapp/
COPY . .
# Fetch dependencies.
# Using go get.
RUN go get -d -v
# Build the binary.
RUN go build -o /go/bin/hello
############################
# STEP 2 build a small image
############################
FROM scratch
# Copy our static executable.
COPY --from=builder /go/bin/hello /go/bin/hello
# Run the hello binary.
ENTRYPOINT ["/go/bin/hello"]
This creates a container image that consists of your application and nothing else,
 making it extremely secure.
```

L'exemple suivant montre également un exemple de construction à plusieurs étapes. Ces types de constructions sont intéressants du point de vue de la sécurité, car vous pouvez les utiliser pour réduire la taille de l'image finale envoyée à votre registre de conteneurs. Les images de conteneur dépourvues d'outils de construction et d'autres fichiers binaires superflus améliorent votre niveau de sécurité en réduisant la surface d'attaque de l'image. Pour plus d'informations sur les versions en plusieurs étapes, voir Multi-stage construit dans le Docker documentation.

Analysez vos images pour détecter les vulnérabilités

À l'instar de leurs homologues machines virtuelles, les images de conteneur peuvent contenir des fichiers binaires et des bibliothèques d'applications présentant des vulnérabilités ou développer des vulnérabilités au fil du temps. Le meilleur moyen de se prémunir contre les attaques est d'analyser régulièrement vos images à l'aide d'un analyseur d'images.

Amazon ECR fournit deux versions d’analyse de base qui utilisent la base de données CVE (Common Vulnerabilities and Exposures) :

- AWS numérisation de base native — Utilisations AWS technologie native, qui est désormais GA et recommandée. Cette analyse de base améliorée est conçue pour fournir aux clients de meilleurs résultats d'analyse et une détection des vulnérabilités sur un large éventail de systèmes d'exploitation courants. Cela permet aux clients de renforcer encore la sécurité des images de leurs conteneurs. Tous les nouveaux registres de clients sont intégrés par défaut à cette version améliorée.
- Numérisation de base Clair : la version de numérisation de base précédente qui utilise le projet open source Clair et est obsolète. Pour plus d'informations sur Clair, voir Clair sur GitHub.

Les deux AWS la numérisation native et la numérisation de base de Clair sont prises en charge dans toutes les régions répertoriées dans AWS Services par région, à l'exception de ceux qui ont été ajoutés après septembre 2024. Le support de Clair étant obsolète, Clair ne sera pas pris en charge dans les nouvelles régions lors de leur ajout et ne sera plus pris en charge dans toutes les régions à compter du 1er octobre 2025.

Amazon ECR utilise la gravité d'un CVE de la source de distribution en amont si disponible. Sinon, le score CVSS (Common Vulnerability Scoring System) est utilisé. Le score CVSS peut être utilisé pour obtenir l'évaluation de gravité de la vulnérabilité NVD. Pour en savoir plus, consultez NVD Vulnerability Gravity Ratings.

Vous pouvez également consulter les résultats d'un scan depuis la console Amazon ECR ou en appelant le DescribeImageScanFindings API. Les images présentant une vulnérabilité `HIGH` ou `CRITICAL` doivent être supprimées ou reconstruites. Si une image déployée développe une vulnérabilité, elle doit être remplacée dès que possible.

Docker Desktop Edge version 2.3.6.0 ou ultérieure peut analyser des images locales. Les analyses sont fournies par Snyk, un service de sécurité des applications. Lorsque des vulnérabilités sont découvertes, Snyk identifie les couches et les dépendances contenant la vulnérabilité dans le Dockerfile. Il recommande également des alternatives sûres, telles que l'utilisation d'une image de base plus fine présentant moins de vulnérabilités ou la mise à niveau d'un package particulier vers une version plus récente. En utilisant l'analyse Docker, les développeurs peuvent résoudre les problèmes de sécurité potentiels avant de transférer leurs images dans le registre.

- Le billet de blog Automating image compliance using Amazon ECR and AWS Security Hub CSPM explique comment intégrer les informations relatives aux vulnérabilités issues d'Amazon ECR dans AWS Security Hub CSPM et automatiser la correction en bloquant l'accès aux images vulnérables.

Supprimez les autorisations spéciales associées à vos images

Les indicateurs des droits d'accès `setuid` et `setgid` autorisent l'exécution d'un exécutable avec les autorisations du propriétaire ou du groupe de l'exécutable. Supprimez de votre image tous les fichiers binaires dotés de ces droits d'accès, car ils peuvent être utilisés pour escalader les privilèges. Envisagez de supprimer tous les shells et utilitaires comme `nc` ou `curl` susceptibles d'être utilisés à des fins malveillantes. Vous pouvez trouver les fichiers avec les droits d'accès `setuid` et `setgid` à l'aide de la commande suivante.

```
find / -perm /6000 -type f -exec ls -ld {} \;
```

Pour supprimer ces autorisations spéciales de ces fichiers, ajoutez la directive suivante à votre image de conteneur.

```
RUN find / -xdev -perm /6000 -type f -exec chmod a-s {} \; || true
```

Créez un ensemble d'images sélectionnées

Plutôt que de permettre aux développeurs de créer leurs propres images, créez un ensemble d'images validées pour les différentes piles d'applications de votre organisation. Ce faisant, les développeurs peuvent renoncer à apprendre à composer des Dockerfiles et se concentrer sur l'écriture de code. Au fur et à mesure que les modifications sont fusionnées dans votre base de code, un CI/CD pipeline peut compiler automatiquement l'actif, puis le stocker dans un référentiel d'artefacts. Enfin, copiez l'artefact dans l'image appropriée avant de le transférer vers un registre Docker tel qu'Amazon ECR. Vous devez créer au minimum un ensemble d'images de base à partir desquelles les développeurs peuvent créer leurs propres Dockerfiles. Vous devez éviter d'extraire des images depuis Docker Hub. Vous ne savez pas toujours ce que contient l'image et celle-ci peut présenter des vulnérabilités.

Analysez les packages d'applications et les bibliothèques pour détecter les vulnérabilités

L'utilisation de bibliothèques open source est désormais courante. Comme les systèmes d'exploitation et les packages de systèmes d'exploitation, ces bibliothèques peuvent présenter des vulnérabilités. Dans le cadre du cycle de développement, ces bibliothèques doivent être analysées et mises à jour lorsque des vulnérabilités critiques sont détectées.

Docker Desktop effectue des analyses locales à l'aide de Snyk. Il peut également être utilisé pour détecter des vulnérabilités et des problèmes de licence potentiels dans les bibliothèques open source. Il peut être intégré directement dans les flux de travail des développeurs, ce qui vous permet d'atténuer les risques posés par les bibliothèques open source. Pour plus d'informations, consultez les rubriques suivantes :

- La page outils de sécurité des applications open source inclut une liste d'outils permettant de détecter les vulnérabilités des applications.

Effectuez une analyse de code statique

Vous devez effectuer une analyse de code statique avant de créer une image de conteneur. Celle-ci est effectuée sur votre code source et est utilisée pour identifier les erreurs de codage et le code susceptible d'être utilisé par un acteur malveillant, comme les injections d'erreurs. Vous pouvez utiliser Amazon Inspector. Pour plus d'informations sur Amazon Inspector, consultez la section Analyse des images de conteneur avec Amazon Inspector dans le Guide de l'utilisateur Amazon Inspector.

Exécutez les conteneurs en tant qu'utilisateur non root

Vous devez exécuter les conteneurs en tant qu'utilisateur non root. Par défaut, les conteneurs s'exécutent en tant qu'utilisateur `root`, sauf si la directive `USER` est incluse dans votre Dockerfile. Les fonctionnalités Linux par défaut attribuées par Docker limitent les actions qui peuvent être exécutées en tant que `root`, mais seulement de manière marginale. Par exemple, un conteneur s'exécutant en tant que `root` n'est toujours pas autorisé à accéder aux appareils.

Dans le cadre de votre CI/CD pipeline, vous devez lancer Dockerfiles pour rechercher le `USER` directive et échouez à la compilation si elle est manquante. Pour plus d'informations, consultez les rubriques suivantes :

- Dockerfile-lint est un outil open source de RedHat qui peut être utilisé pour vérifier si le fichier est conforme aux meilleures pratiques.
- Hadolint est un autre outil permettant de créer des images Docker conformes aux bonnes pratiques.

Utilisez un système de fichiers racine en lecture seule

Vous devez utiliser un système de fichiers racine en lecture seule. Le système de fichiers racine d'un conteneur est accessible en écriture par défaut. Lorsque vous configurez un conteneur avec un système de fichiers racine `RO` (en lecture seule), cela vous oblige à définir explicitement où les données peuvent être conservées. Cela réduit votre surface d'attaque, car le système de fichiers du conteneur ne peut pas être disponible en écriture à moins que des autorisations ne soient spécifiquement accordées.

> **Note**
>
> Le fait d'avoir un système de fichiers racine en lecture seule peut entraîner des problèmes avec certains packages du système d'exploitation qui devraient pouvoir écrire dans le système de fichiers. Si vous prévoyez d'utiliser des systèmes de fichiers racine en lecture seule, effectuez des tests approfondis au préalable.

Configurez des tâches avec des limites de processeur et de mémoire (Amazon EC2)

Vous devez configurer les tâches avec des limites de processeur et de mémoire afin de réduire les risques suivants. Les limites de ressources d'une tâche définissent une limite supérieure pour la quantité de processeur et de mémoire pouvant être réservée par tous les conteneurs d'une tâche. Si aucune limite n'est définie, les tâches ont accès au processeur et à la mémoire de l'hôte. Cela peut entraîner des problèmes, car les tâches déployées sur un hôte partagé peuvent priver d'autres tâches de ressources système.

> **Note**
>
> Amazon ECS sur AWS Fargate les tâches nécessitent que vous spécifiiez des limites de processeur et de mémoire car elles utilisent ces valeurs à des fins de facturation. Une seule tâche monopolisant toutes les ressources du système n'est pas un problème pour Amazon ECS Fargate, car chaque tâche est exécutée sur sa propre instance dédiée. Si vous ne spécifiez pas de limite de mémoire, Amazon ECS alloue un minimum de 4 Mo à chaque conteneur. De même, si aucune limite de processeur n'est définie pour la tâche, l'agent de conteneur Amazon ECS lui attribue un minimum de deux processeurs.

Utilisez des balises immuables avec Amazon ECR

Avec Amazon ECR, vous pouvez et devez utiliser des images configurées avec des balises immuables. Cela empêche de transférer une version modifiée ou mise à jour d'une image vers votre référentiel d'images avec une balise identique. Cela permet d'éviter qu'un attaquant ne diffuse une version compromise d'une image sur votre image portant la même balise. En utilisant des balises immuables, vous vous forcez efficacement à envoyer une nouvelle image avec une balise différente pour chaque modification.

Évitez d'exécuter des conteneurs dotés de privilèges (Amazon EC2)

Vous devez éviter d'exécuter des conteneurs dotés de privilèges. En effet, les conteneurs s'exécutant comme `privileged` sont exécutés avec des privilèges étendus sur l'hôte. Cela signifie que le conteneur hérite de toutes les fonctionnalités Linux attribuées à `root` sur l'hôte. Son utilisation doit être strictement restreinte ou interdite. Nous vous conseillons de définir la variable d'environnement de l'agent de conteneur Amazon ECS `ECS_DISABLE_PRIVILEGED` sur `true` afin d'empêcher les conteneurs de s'exécuter comme `privileged` sur des hôtes particuliers si `privileged` n'est pas nécessaire. Vous pouvez également utiliser AWS Lambda pour scanner vos définitions de tâches à des fins d'utilisation du `privileged` paramètre.

> **Note**
>
> L'exécution d'un conteneur en tant que `privileged` n'est pas prise en charge sur Amazon ECS sur AWS Fargate.

Supprimez les fonctionnalités Linux inutiles du conteneur

Voici une liste des fonctionnalités Linux par défaut attribuées aux conteneurs Docker. Pour plus d'informations sur chaque fonctionnalité, veuillez consulter Présentation des fonctionnalités de Linux (langue française non garantie).

```
CAP_CHOWN, CAP_DAC_OVERRIDE, CAP_FOWNER, CAP_FSETID, CAP_KILL,
CAP_SETGID, CAP_SETUID, CAP_SETPCAP, CAP_NET_BIND_SERVICE,
CAP_NET_RAW, CAP_SYS_CHROOT, CAP_MKNOD, CAP_AUDIT_WRITE,
CAP_SETFCAP
```

Si un conteneur ne nécessite pas toutes les fonctionnalités de noyau Docker répertoriées ci-dessus, pensez à les supprimer du conteneur. Pour plus d'informations sur chaque fonctionnalité du noyau Docker, consultez KernelCapabilities. Vous pouvez déterminer les fonctionnalités utilisées en procédant ainsi :

- Installez le package du système d'exploitation libcap-ng et exécutez l'utilitaire `pscap` pour répertorier les fonctionnalités utilisées par chaque processus.
- Vous pouvez également utiliser capsh pour déchiffrer les fonctionnalités utilisées par un processus.

Utilisez une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR

Vous devez utiliser une clé gérée par le client (CMK) pour chiffrer les images transférées vers Amazon ECR. Les images transmises à Amazon ECR sont automatiquement cryptées au repos à l'aide d'un AWS Key Management Service (AWS KMS) clé gérée. Si vous préférez utiliser votre propre clé, Amazon ECR prend désormais en charge AWS KMS chiffrement à l'aide de clés gérées par le client (CMK). Avant d'activer le chiffrement côté serveur avec une clé CMK, consultez les considérations répertoriées dans la documentation sur le chiffrement au repos.

Didacticiels pour Amazon ECS

Les didacticiels suivants montrent comment exécuter les tâches courantes à l'aide d'Amazon ECS.

Vous pouvez utiliser l’un des didacticiels suivants pour en savoir plus sur la prise en main d’Amazon ECS.

Aperçu du didacticiel	En savoir plus
Démarrage avec Amazon ECS sur Fargate.	Création d’une tâche Amazon ECS Linux pour Fargate
Démarrage avec les conteneurs Windows sur Fargate.	Création d’une tâche Windows Amazon ECS pour Fargate
Démarrage avec les conteneurs Windows sur EC2.	Création d’une tâche Windows Amazon ECS pour EC2

Vous pouvez utiliser l'un des didacticiels suivants pour déployer des tâches sur Amazon ECS à l'aide du AWS CLI

Aperçu du didacticiel	En savoir plus
Création d’une tâche Linux pour Fargate.	Création d'une tâche Linux Amazon ECS pour le Fargate avec AWS CLI
Création d’une tâche Windows pour Fargate.	Création d'une tâche Windows Amazon ECS pour le Fargate avec AWS CLI
Création d’une tâche Linux pour EC2.	Création d'une tâche Amazon ECS pour EC2 avec AWS CLI

Vous pouvez utiliser l’un des didacticiels suivants pour en savoir plus sur la surveillance et la journalisation.

Aperçu du didacticiel	En savoir plus
Configurez une fonction Lambda simple qui écoute les événements des tâches et les écrit dans un flux de CloudWatch journal des journaux.	Configuration d'Amazon ECS pour écouter CloudWatch les événements
Configurez une règle d' EventBridge événement Amazon qui capture uniquement les événement s de tâche lorsque la tâche a cessé de s'exécuter parce que l'un de ses conteneurs essentiels s'est arrêté.	Envoi d’alertes Amazon Simple Notification Service pour les événements de tâche Amazon ECS arrêtés
Configuration de messages de journaux qui appartiennent à l’origine à un seul contexte, mais qui ont été divisés en plusieurs enregistrements ou lignes de journal.	Concaténation des messages du journal Amazon ECS multiligne ou de suivi de pile
Déployez des conteneurs Fluent Bit sur leurs instances Windows exécutées dans Amazon ECS pour diffuser les journaux générés par les tâches Windows vers Amazon afin CloudWatch de centraliser la journalisation.	Déploiement de Fluent Bit sur Amazon ECS pour des conteneurs Windows

Vous pouvez utiliser l'un des didacticiels suivants pour en savoir plus sur l'utilisation de l'authentification Active Directory avec un compte de service géré de groupe sur Amazon ECS.

Aperçu du didacticiel	En savoir plus
Utilisation d'un compte de service géré de groupe avec des conteneurs Linux sur EC2.	Utilisation gMSA pour EC2 Linux conteneurs sur Amazon ECS
Utilisation d'un compte de service géré de groupe avec des conteneurs Windows sur EC2.	Utilisation des gMSA pour les conteneurs Windows EC2 pour Amazon ECS
Utilisation d'un compte de service géré de groupe avec des conteneurs Linux sur Fargate.	Utilisation gMSA for Linux conteneurs sur Fargate
Création d'une tâche qui exécute un conteneur Windows disposant des informations d'identification nécessaires pour accéder à Active Directory avec un compte de service géré de groupe sans domaine.	Utilisation de conteneur s Windows Amazon ECS avec un environnement sans domaine gMSA en utilisant le AWS CLI

Création d'une tâche Linux Amazon ECS pour le Fargate avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche Linux et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Conditions préalables
- Étape 1 : Créer un cluster
- Étape 2 : Enregistrer une définition de tâche Linux
- Étape 3 : Répertorier les définitions de tâche
- Étape 4 : Créer un service
- Étape 5 : Répertorier les services
- Étape 6 : Décrire le service en cours d'exécution
- Étape 7 : Test
- Étape 8 : Nettoyer

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants.

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI, Installation ou mise à jour vers la dernière version du AWS CLI.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l’exemple de politique IAM Amazon ECS_FullAccess.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Amazon ECR Public afin que votre tâche ait accès à Internet. Pour donner à votre tâche un itinéraire vers Internet, utilisez l'une des options suivantes.
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et affectez une adresse IP publique à la tâche.

Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.

Pour de plus amples informations sur les groupes de sécurité et les règles, veuillez consulter Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

- Si vous suivez ce didacticiel en utilisant un sous-réseau privé, vous pouvez utiliser Amazon ECS Exec pour interagir directement avec votre conteneur et tester le déploiement. Vous devez créer un rôle IAM de tâche pour utiliser ECS Exec. Pour plus d’informations sur le rôle IAM de tâche et les autres prérequis, consultez la section Surveillance des conteneurs Amazon ECS avec Amazon ECS Exec.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : Créer un cluster

Par défaut, votre compte reçoit un cluster `default`.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Sortie :

```
{
    "cluster": {
        "status": "ACTIVE",
        "defaultCapacityProviderStrategy": [],
```

```
        "statistics": [],
        "capacityProviders": [],
        "tags": [],
        "clusterName": "fargate-cluster",
        "settings": [
            {
                "name": "containerInsights",
                "value": "disabled"
            }
        ],
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Étape 2 : Enregistrer une définition de tâche Linux

Avant de pouvoir exécuter une tâche sur votre cluster ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une définition de tâche simple qui crée une application Web PHP à l'aide de l'image de conteneur httpd hébergée sur Docker Hub. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS. Pour ce didacticiel, l'`taskRoleArn` n'est nécessaire que si vous déployez la tâche dans un sous-réseau privé et souhaitez tester le déploiement. Remplacez l'`taskRoleArn` par le rôle de tâche IAM que vous avez créé pour utiliser ECS Exec, comme indiqué dans Conditions préalables.

```
{
        "family": "sample-fargate",
        "networkMode": "awsvpc",
        "taskRoleArn": "arn:aws:iam::aws_account_id:role/execCommandRole",
        "containerDefinitions": [
            {
                "name": "fargate-app",
                "image": "public.ecr.aws/docker/library/httpd:latest",
                "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\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

Enregistrez la définition de tâche JSON sous forme de fichier et transmettez-la avec l'option `--cli-input-json file://path_to_file.json`.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 3 : Répertorier les définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez run-task ou start-task.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate:1"
    ]
}
```

Étape 4 : Créer un service

Une fois que vous avez enregistré une tâche pour votre compte, vous pouvez créer un service pour la tâche enregistrée dans votre cluster. Pour cet exemple, vous créez un service avec une instance de la définition de tâche `sample-fargate:1` exécutée dans votre cluster. La tâche nécessite un itinéraire vers Internet. Il y a deux façons d'y parvenir. Une façon consiste à utiliser un sous-réseau privé configuré avec une passerelle NAT avec une adresse IP Elastic dans un sous-réseau public. Une autre façon consiste à utiliser un sous-réseau public et à attribuer une adresse IP publique à votre tâche. Nous fournissons les deux exemples dans les sections suivantes.

Exemple d'utilisation d'un sous-réseau privé. L'option `enable-execute-command` est nécessaire pour utiliser Amazon ECS Exec.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234]}" --enable-execute-command
```

Exemple d'utilisation d'un sous-réseau public.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service --
task-definition sample-fargate:1 --desired-count 1 --launch-type "FARGATE" --network-
configuration "awsvpcConfiguration={subnets=[subnet-abcd1234],securityGroups=[sg-
abcd1234],assignPublicIp=ENABLED}"
```

La commande create-service renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 5 : Répertorier les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter le nom du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster fargate-cluster
```

Sortie :

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
    ]
}
```

Étape 6 : Décrire le service en cours d'exécution

Décrivez le service à l'aide du nom de service récupéré précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

En cas de réussite, une description des défaillances de service et des services est renvoyée. Par exemple, dans la section `services`, vous trouverez des informations sur les déploiements, telles que le statut des tâches en cours d'exécution ou en attente. Vous trouverez également des informations sur la définition des tâches, la configuration réseau et les événements horodatés. Dans la section des défaillances, vous trouverez des informations sur les défaillances, le cas échéant, associées à l'appel. Pour le dépannage, consultez Messages d'événements de service. Pour de plus amples informations sur la description du service, consultez Description des services.

```
{
    "services": [
        {
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
```

```
                    "sg-abcd1234"
                ],
                "assignPublicIp": "ENABLED"
            }
        },
        "launchType": "FARGATE",
        "enableECSManagedTags": false,
        "loadBalancers": [],
        "deploymentController": {
            "type": "ECS"
        },
        "desiredCount": 1,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
        "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/
fargate-service",
        "deploymentConfiguration": {
            "maximumPercent": 200,
            "minimumHealthyPercent": 100
        },
        "createdAt": 1692283199.771,
        "schedulingStrategy": "REPLICA",
        "placementConstraints": [],
        "deployments": [
            {
                "status": "PRIMARY",
                "networkConfiguration": {
                    "awsvpcConfiguration": {
                        "subnets": [
                            "subnet-abcd1234"
                        ],
                        "securityGroups": [
                            "sg-abcd1234"
                        ],
                        "assignPublicIp": "ENABLED"
                    }
                },
                "pendingCount": 0,
                "launchType": "FARGATE",
                "createdAt": 1692283199.771,
                "desiredCount": 1,
                "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate:1",
                "updatedAt": 1692283199.771,
                "platformVersion": "1.4.0",
```

```
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "serviceName": "fargate-service",
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "status": "ACTIVE",
            "serviceRegistries": [],
            "pendingCount": 0,
            "createdBy": "arn:aws:iam::aws_account_id:user/user_name",
            "platformVersion": "LATEST",
            "placementStrategy": [],
            "propagateTags": "NONE",
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate:1"
        }
    ],
    "failures": []
}
```

Étape 7 : Test

Tâche de test déployée à l'aide d'un sous-réseau public

Décrivez la tâche dans le service afin que vous puissiez obtenir l'interface réseau Elastic (ENI) pour la tâche.

Tout d'abord, obtenez l'ARN de la tâche.

```
aws ecs list-tasks --cluster fargate-cluster --service fargate-service
```

La sortie contient l'ARN de la tâche.

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE
    ]
}
```

Décrivez la tâche et localisez l'ID de l'ENI. Utilisez l'ARN de la tâche pour le paramètre `tasks`.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/service/EXAMPLE
```

Les informations concernant la pièce jointe sont répertoriées dans la sortie.

```
{
    "tasks": [
        {
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
```

```
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                    ]
                }
…
}
```

Décrivez l'ENI pour obtenir l'adresse IP publique.

```
aws ec2 describe-network-interfaces --network-interface-id  eni-0fa40520aeEXAMPLE
```

L'adresse IP publique se trouve dans la sortie.

```
{
    "NetworkInterfaces": [
        {
            "Association": {
                "IpOwnerId": "amazon",
                "PublicDnsName": "ec2-34-229-42-222.compute-1.amazonaws.com",
                "PublicIp": "198.51.100.2"
            },
…
}
```

Saisissez l'adresse IP publique dans votre navigateur web. Vous devriez voir une page web qui affiche l'exemple d'application Amazon ECS.

Tâche de test déployée à l'aide d'un sous-réseau privé

Décrivez la tâche et localisez `managedAgents` pour vérifier que `ExecuteCommandAgent` est en cours d'exécution. Notez `privateIPv4Address` pour une utilisation ultérieure.

```
aws ecs describe-tasks --cluster fargate-cluster --tasks arn:aws:ecs:us-
east-1:123456789012:task/fargate-service/EXAMPLE
```

Les informations concernant l'agent géré sont répertoriées dans la sortie.

```
{
     "tasks": [
        {
```

```
            "attachments": [
                {
                    "id": "d9e7735a-16aa-4128-bc7a-b2d5115029e9",
                    "type": "ElasticNetworkInterface",
                    "status": "ATTACHED",
                    "details": [
                        {
                            "name": "subnetId",
                            "value": "subnetabcd1234"
                        },
                        {
                            "name": "networkInterfaceId",
                            "value": "eni-0fa40520aeEXAMPLE"
                        },
                        {
                            "name": "privateIPv4Address",
                            "value": "10.0.143.156"
                        }
                    ]
                }
            ],
        ...
        "containers": [
            {
            ...
           "managedAgents": [
                        {
                            "lastStartedAt": "2023-08-01T16:10:13.002000+00:00",
                            "name": "ExecuteCommandAgent",
                            "lastStatus": "RUNNING"
                        }
                ],
            ...
    }
```

Après avoir vérifié que `ExecuteCommandAgent` est en cours d'exécution, vous pouvez exécuter la commande suivante pour exécuter un shell interactif sur le conteneur de la tâche.

```
aws ecs execute-command --cluster fargate-cluster \
    --task  arn:aws:ecs:us-east-1:123456789012:task/fargate-service/EXAMPLE  \
    --container  fargate-app \
    --interactive \
    --command "/bin/sh"
```

Une fois le shell interactif lancé, exécutez les commandes suivantes pour installer cURL.

```
apt update
```

```
apt install curl
```

Après avoir installé cURL, exécutez la commande suivante en utilisant l'adresse IP privée que vous avez obtenue précédemment.

```
 curl 10.0.143.156
```

Vous devriez voir l'équivalent HTML de la page Web d'exemple d'application Amazon ECS.

```
<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>
```

Étape 8 : Nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Supprimez le service.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Supprimez le cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Création d'une tâche Windows Amazon ECS pour le Fargate avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche Windows et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Veillez à utiliser la dernière version de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Conditions préalables
- Étape 1 : Créer un cluster
- Étape 2 : Enregistrement d'une définition de tâche Windows
- Étape 3 : Répertorier les définitions de tâche
- Étape 4 : Créer un service
- Étape 5 : Répertorier les services
- Étape 6 : Décrire le service en cours d'exécution
- Étape 7 : Nettoyer

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants.

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS CLI.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.

- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l'exemple de politique IAM Amazon ECS_FullAccess.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Ce didacticiel utilise une image de conteneur hébergée sur Docker Hub afin que votre tâche ait accès à Internet. Pour donner à votre tâche un itinéraire vers Internet, utilisez l'une des options suivantes.
  - Utilisez un sous-réseau privé avec une passerelle NAT dotée d'une adresse IP Elastic.
  - Utilisez un sous-réseau public et affectez une adresse IP publique à la tâche.

  Pour de plus amples informations, veuillez consulter the section called "Créer un Virtual Private Cloud".

  Pour de plus amples informations sur les groupes de sécurité et les règles, veuillez consulter Groupes de sécurité par défaut de vos VPC et Exemples de règles dans le Guide de l'utilisateur Amazon Virtual Private Cloud.
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : Créer un cluster

Par défaut, votre compte reçoit un cluster `default`.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name fargate-cluster
```

Sortie :

```
{
```

```
    "cluster": {
        "status": "ACTIVE",
        "statistics": [],
        "clusterName": "fargate-cluster",
        "registeredContainerInstancesCount": 0,
        "pendingTasksCount": 0,
        "runningTasksCount": 0,
        "activeServicesCount": 0,
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster"
    }
}
```

Étape 2 : Enregistrement d'une définition de tâche Windows

Avant de pouvoir exécuter une tâche Windows sur votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L'exemple suivant est une simple définition de tâche qui crée une application web. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

```
{
    "containerDefinitions": [
        {
            "command": ["New-Item -Path C:\\inetpub\\wwwroot\\index.html -Type file
 -Value '<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>'; C:\\ServiceMonitor.exe
 w3svc"],
            "entryPoint": [
                "powershell",
                "-Command"
            ],
            "essential": true,
            "cpu": 2048,
            "memory": 4096,
            "image": "mcr.microsoft.com/windows/servercore/iis:windowsservercore-
ltsc2019",
            "name": "sample_windows_app",
            "portMappings": [
                {
                    "hostPort": 80,
```

```
                    "containerPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "memory": "4096",
    "cpu": "2048",
    "networkMode": "awsvpc",
    "family": "windows-simple-iis-2019-core",
    "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
    "runtimePlatform": {"operatingSystemFamily": "WINDOWS_SERVER_2019_CORE"},
    "requiresCompatibilities": ["FARGATE"]
}
```

L'exemple JSON précédent peut être transmis AWS CLI au de deux manières : vous pouvez enregistrer la définition de tâche JSON sous forme de fichier et la transmettre avec l'`--cli-input-json file://`*`path_to_file.json`*option.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/fargate-task.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 3 : Répertorier les définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez run-task ou start-task.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
        "arn:aws:ecs:region:aws_account_id:task-definition/sample-fargate-windows:1"
    ]
}
```

Étape 4 : Créer un service

Une fois que vous avez enregistré une tâche pour votre compte, vous pouvez créer un service pour la tâche enregistrée dans votre cluster. Pour cet exemple, vous créez un service avec une instance de la définition de tâche `sample-fargate:1` exécutée dans votre cluster. La tâche nécessite un itinéraire vers Internet. Il y a deux façons d'y parvenir. Une façon consiste à utiliser un sous-réseau privé configuré avec une passerelle NAT avec une adresse IP Elastic dans un sous-réseau public. Une autre façon consiste à utiliser un sous-réseau public et à attribuer une adresse IP publique à votre tâche. Nous fournissons les deux exemples dans les sections suivantes.

Exemple d'utilisation d'un sous-réseau privé.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234]}"
```

Exemple d'utilisation d'un sous-réseau public.

```
aws ecs create-service --cluster fargate-cluster --service-name fargate-service
 --task-definition sample-fargate-windows:1 --desired-count 1 --launch-type
 "FARGATE" --network-configuration "awsvpcConfiguration={subnets=[subnet-
abcd1234],securityGroups=[sg-abcd1234],assignPublicIp=ENABLED}"
```

La commande create-service renvoie une description de la définition de tâche une fois l'enregistrement terminé.

Étape 5 : Répertorier les services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter le nom du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster fargate-cluster
```

Sortie :

```
{
    "serviceArns": [
        "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/fargate-service"
```

```
    ]
}
```

Étape 6 : Décrire le service en cours d'exécution

Décrivez le service à l'aide du nom de service récupéré précédemment afin d'obtenir plus d'informations sur la tâche.

```
aws ecs describe-services --cluster fargate-cluster --services fargate-service
```

En cas de réussite, une description des défaillances de service et des services est renvoyée. Par exemple, dans la section des services, vous trouverez des informations sur les déploiements, telles que le statut des tâches en cours d'exécution ou en attente. Vous trouverez également des informations sur la définition des tâches, la configuration réseau et les événements horodatés. Dans la section des défaillances, vous trouverez des informations sur les défaillances, le cas échéant, associées à l'appel. Pour le dépannage, consultez Messages d'événements de service. Pour de plus amples informations sur la description du service, consultez Description des services.

```
{
    "services": [
        {
            "status": "ACTIVE",
            "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-definition/
sample-fargate-windows:1",
            "pendingCount": 2,
            "launchType": "FARGATE",
            "loadBalancers": [],
            "roleArn": "arn:aws:iam::aws_account_id:role/aws-service-role/
ecs.amazonaws.com/AWSServiceRoleForECS",
            "placementConstraints": [],
            "createdAt": 1510811361.128,
            "desiredCount": 2,
            "networkConfiguration": {
                "awsvpcConfiguration": {
                    "subnets": [
                        "subnet-abcd1234"
                    ],
                    "securityGroups": [
                        "sg-abcd1234"
                    ],
                    "assignPublicIp": "DISABLED"
```

```
                }
            },
            "platformVersion": "LATEST",
            "serviceName": "fargate-service",
            "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/fargate-cluster",
            "serviceArn": "arn:aws:ecs:region:aws_account_id:service/fargate-cluster/
fargate-service",
            "deploymentConfiguration": {
                "maximumPercent": 200,
                "minimumHealthyPercent": 100
            },
            "deployments": [
                {
                    "status": "PRIMARY",
                    "networkConfiguration": {
                        "awsvpcConfiguration": {
                            "subnets": [
                                "subnet-abcd1234"
                            ],
                            "securityGroups": [
                                "sg-abcd1234"
                            ],
                            "assignPublicIp": "DISABLED"
                        }
                    },
                    "pendingCount": 2,
                    "launchType": "FARGATE",
                    "createdAt": 1510811361.128,
                    "desiredCount": 2,
                    "taskDefinition": "arn:aws:ecs:region:aws_account_id:task-
definition/sample-fargate-windows:1",
                    "updatedAt": 1510811361.128,
                    "platformVersion": "0.0.1",
                    "id": "ecs-svc/9223370526043414679",
                    "runningCount": 0
                }
            ],
            "events": [
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 53c0de40-ea3b-489f-a352-623bf1235f08) (task d0aec985-901b-488f-9fb4-61b991b332a3).",
                    "id": "92b8443e-67fb-4886-880c-07e73383ea83",
                    "createdAt": 1510811841.408
                },
```

```
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 b4911bee-7203-4113-99d4-e89ba457c626) (task cc5853e3-6e2d-4678-8312-74f8a7d76474).",
                    "id": "d85c6ec6-a693-43b3-904a-a997e1fc844d",
                    "createdAt": 1510811601.938
                },
                {
                    "message": "(service fargate-service) has started 2 tasks: (task
 cba86182-52bf-42d7-9df8-b744699e6cfc) (task f4c1ad74-a5c6-4620-90cf-2aff118df5fc).",
                    "id": "095703e1-0ca3-4379-a7c8-c0f1b8b95ace",
                    "createdAt": 1510811364.691
                }
            ],
            "runningCount": 0,
            "placementStrategy": []
        }
    ],
    "failures": []
}
```

Étape 7 : Nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Supprimez le service.

```
aws ecs delete-service --cluster fargate-cluster --service fargate-service --force
```

Supprimez le cluster.

```
aws ecs delete-cluster --cluster fargate-cluster
```

Création d'une tâche Amazon ECS pour EC2 avec AWS CLI

Les étapes suivantes vous aideront à configurer un cluster, enregistrer une définition de tâche, exécuter une tâche et effectuer d'autres scénarios courants dans Amazon ECS avec la AWS CLI. Utilisez la version la plus récente de l' AWS CLI. Pour savoir comment opérer une mise à niveau vers la dernière version, consultez la section Installation ou mise à jour vers la dernière version de l' AWS CLI.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Rubriques

- Conditions préalables
- Créer un cluster
- Lancement d’une instance de conteneur avec l’AMI Amazon ECS
- Liste des instances de conteneur
- Description de votre instance de conteneur
- Enregistrement d’une définition de tâche
- Liste des définitions de tâche
- Créer un service
- Liste des services
- Description des services
- Description de la tâche en cours d’exécution
- Test du service Web
- nettoyer des ressources ;

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS CLI.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l’exemple de politique IAM Amazon ECS_FullAccess.
- Un rôle IAM d’instance de conteneur a été créé pour être utilisé. Pour en savoir plus, consultezRôle IAM d'instance de conteneur Amazon ECS.

- Vous disposez d'un VPC créé pour être utilisé. Pour de plus amples informations, veuillez consulter the section called "Créer un Virtual Private Cloud".
- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Créer un cluster

Par défaut, votre compte reçoit un cluster `default` lorsque vous lancez votre première instance de conteneur.

> **Note**
>
> L'utilisation du cluster `default` qui vous est fourni présente l'avantage que vous n'avez pas besoin de spécifier l'option `--cluster` *`cluster_name`* dans les commandes suivantes. Si vous créez votre propre cluster, sans valeur par défaut, vous devez spécifier `--cluster` *`cluster_name`* pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Créez votre propre cluster avec un nom unique à l'aide de la commande suivante :

```
aws ecs create-cluster --cluster-name MyCluster
```

Sortie :

```
{
    "cluster": {
        "clusterName": "MyCluster",
        "status": "ACTIVE",
        "clusterArn": "arn:aws:ecs:region:aws_account_id:cluster/MyCluster"
    }
}
```

Lancement d'une instance de conteneur avec l'AMI Amazon ECS

Les instances de conteneur sont des instances EC2 qui exécutent l'agent de conteneur Amazon ECS et ont été enregistrées dans un cluster. Dans cette section, vous allez lancer une instance EC2 à l'aide de l' ECS-optimized AMI.

Pour lancer une instance de conteneur avec le AWS CLI

1. Récupérez l'ID AMI ECS-optimized Amazon Linux 2 le plus récent pour vous Région AWS à l'aide de la commande suivante. Cette commande utilise le AWS Systems Manager Parameter Store pour obtenir le dernier ID d' ECS-optimizedAMI. L'AMI inclut l'agent de conteneur Amazon ECS et l'exécution Docker préinstallés.

   ```
   aws ssm get-parameters --names /aws/service/ecs/optimized-ami/amazon-linux-2/
   recommended --query 'Parameters[0].Value' --output text | jq -r '.image_id'
   ```

   Sortie :

   ```
   ami-abcd1234
   ```

2. Créez un groupe de sécurité qui autorise l'accès SSH pour gérer votre instance de conteneur et l'accès HTTP pour le serveur Web.

   ```
   aws ec2 create-security-group --group-name ecs-tutorial-sg --description "ECS
    tutorial security group"
   ```

   Sortie :

   ```
   {
       "GroupId": "sg-abcd1234"
   }
   ```

3. Ajoutez une règle entrante au groupe de sécurité en exécutant la commande suivante.

   ```
   aws ec2 authorize-security-group-ingress --group-id sg-abcd1234 --protocol tcp --
   port 80 --cidr 0.0.0.0/0
   ```

   Sortie :

   ```
   {
       "Return": true,
       "SecurityGroupRules": [
           {
               "SecurityGroupRuleId": "sgr-efgh5678",
               "GroupId": "sg-abcd1234",
               "GroupOwnerId": "123456789012",
   ```

```
            "IsEgress": false,
            "IpProtocol": "tcp",
            "FromPort": 80,
            "ToPort": 80,
            "CidrIpv4": "0.0.0.0/0"
        }
    ]
}
```

Le groupe de sécurité autorise désormais l'accès SSH depuis la plage d'adresses IP spécifiée et l'accès HTTP depuis n'importe où. Dans un environnement de production, vous devez restreindre l'accès SSH à votre adresse IP spécifique et envisager de limiter l'accès HTTP selon les besoins.

4. Créez une paire de clés EC2 pour l'accès SSH à votre instance de conteneur.

```
aws ec2 create-key-pair --key-name ecs-tutorial-key --query 'KeyMaterial' --output
 text > ecs-tutorial-key.pem
chmod 400 ecs-tutorial-key.pem
```

La clé privée est enregistrée sur votre ordinateur local avec les autorisations d'accès SSH appropriées.

5. Lancez une instance EC2 à l'aide de l' ECS-optimized AMI et configurez-la pour qu'elle rejoigne votre cluster.

```
aws ec2 run-instances --image-id ami-abcd1234 --instance-type t3.micro --key-
name ecs-tutorial-key --security-group-ids sg-abcd1234 --iam-instance-profile
 Name=ecsInstanceRole --user-data '#!/bin/bash
echo ECS_CLUSTER=MyCluster >> /etc/ecs/ecs.config'
{
    "Instances": [
        {
            "InstanceId": "i-abcd1234",
            "ImageId": "ami-abcd1234",
            "State": {
                "Code": 0,
                "Name": "pending"
            },
            "PrivateDnsName": "",
            "PublicDnsName": "",
            "StateReason": {
```

```
                "Code": "pending",
                "Message": "pending"
            },
            "InstanceType": "t3.micro",
            "KeyName": "ecs-tutorial-key",
            "LaunchTime": "2025-01-13T10:30:00.000Z"
        }
    ]
}
```

Le script de données utilisateur configure l'agent Amazon ECS pour enregistrer l'instance auprès de votre `MyCluster`. L'instance utilise le rôle IAM `ecsInstanceRole`, qui fournit les autorisations nécessaires à l'agent.

Liste des instances de conteneur

Quelques minutes après le lancement de votre instance de conteneur, l'agent Amazon ECS enregistre l'instance auprès de votre MyCluster cluster. Vous pouvez afficher les instances de conteneur d'un cluster en exécutant la commande suivante :

```
aws ecs list-container-instances --cluster MyCluster
```

Sortie :

```
{
    "containerInstanceArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:container-instance/container_instance_ID"
    ]
}
```

Description de votre instance de conteneur

Après avoir obtenu l'ARN ou l'ID d'une instance de conteneur, vous pouvez utiliser la commande describe-container-instances pour obtenir des informations importantes sur l'instance, comme les ressources UC et de mémoire restantes et enregistrées.

```
aws ecs describe-container-instances --cluster MyCluster --container-
instances container_instance_ID
```

Sortie :


```
{
    "failures": [],
    "containerInstances": [
        {
            "status": "ACTIVE",
            "registeredResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
```

```
            "ec2InstanceId": "instance_id",
            "agentConnected": true,
            "containerInstanceArn": "arn:aws:ecs:us-west-2:aws_account_id:container-
instance/container_instance_ID",
            "pendingTasksCount": 0,
            "remainingResources": [
                {
                    "integerValue": 1024,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "CPU",
                    "doubleValue": 0.0
                },
                {
                    "integerValue": 995,
                    "longValue": 0,
                    "type": "INTEGER",
                    "name": "MEMORY",
                    "doubleValue": 0.0
                },
                {
                    "name": "PORTS",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [
                        "22",
                        "2376",
                        "2375",
                        "51678"
                    ],
                    "type": "STRINGSET",
                    "integerValue": 0
                },
                {
                    "name": "PORTS_UDP",
                    "longValue": 0,
                    "doubleValue": 0.0,
                    "stringSetValue": [],
                    "type": "STRINGSET",
                    "integerValue": 0
                }
            ],
            "runningTasksCount": 0,
            "attributes": [
```

```
                {
                    "name": "com.amazonaws.ecs.capability.privileged-container"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.17"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.18"
                },
                {
                    "name": "com.amazonaws.ecs.capability.docker-remote-api.1.19"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.json-file"
                },
                {
                    "name": "com.amazonaws.ecs.capability.logging-driver.syslog"
                }
            ],
            "versionInfo": {
                "agentVersion": "1.5.0",
                "agentHash": "b197edd",
                "dockerVersion": "DockerVersion: 1.7.1"
            }
        }
    ]
}
```

Vous pouvez également trouver l'ID d'instance Amazon EC2 que vous pouvez utiliser pour surveiller l'instance dans la console Amazon EC2 ou avec la commande aws ec2 describe-instances --instance-id ***instance_id***.

Enregistrement d’une définition de tâche

Avant de pouvoir exécuter une tâche sur votre cluster Amazon ECS, vous devez enregistrer une définition de tâche. Les définitions de tâches sont des listes de conteneurs regroupés ensemble. L’exemple suivant est une définition de tâche simple qui utilise une image `nginx`. Pour plus d'informations sur les paramètres de définition des tâches disponibles, consultez Définitions de tâche Amazon ECS.

```
{
    "family": "nginx-task",
```

```
    "containerDefinitions": [
        {
            "name": "nginx",
            "image": "public.ecr.aws/ecs-sample-image/amazon-ecs-sample:latest",
            "cpu": 256,
            "memory": 512,
            "essential": true,
            "portMappings": [
                {
                    "containerPort": 80,
                    "hostPort": 80,
                    "protocol": "tcp"
                }
            ]
        }
    ],
    "requiresCompatibilities": ["EC2"],
    "networkMode": "bridge"
}
```

L'exemple JSON précédent peut être transmis AWS CLI au de deux manières : vous pouvez enregistrer la définition de tâche JSON sous forme de fichier et la transmettre avec l'`--cli-input-json file://`*`path_to_file.json`*option. Vous pouvez également utiliser un caractère d'échappement pour les guillemets dans le fichier JSON et transmettre les définitions de conteneur JSON sur la ligne de commande. Si vous choisissez de transmettre les définitions de conteneur sur la ligne de commande, votre commande nécessite également un paramètre `--family` qui permet de conserver plusieurs versions de votre définition de tâche associées les unes aux autres.

Pour utiliser un fichier JSON pour les définitions de conteneur :

```
aws ecs register-task-definition --cli-input-json file://$HOME/tasks/nginx.json
```

La commande register-task-definition renvoie une description de la définition de tâche une fois l'enregistrement terminé.

```
{
    "taskDefinition": {
        "taskDefinitionArn": "arn:aws:ecs:us-east-1:123456789012:task-definition/nginx-
task:1",
        "family": "nginx-task",
        "revision": 1,
        "status": "ACTIVE",
```

```
        "containerDefinitions": [
            {
                "name": "nginx",
                "image": "public.ecr.aws/docker/library/nginx:latest",
                "cpu": 256,
                "memory": 512,
                "essential": true,
                "portMappings": [
                    {
                        "containerPort": 80,
                        "hostPort": 80,
                        "protocol": "tcp"
                    }
                ],
                "environment": [],
                "mountPoints": [],
                "volumesFrom": []
            }
        ],
        "volumes": [],
        "networkMode": "bridge",
        "compatibilities": [
            "EC2"
        ],
        "requiresCompatibilities": [
            "EC2"
        ]
    }
}
```

Liste des définitions de tâche

Vous pouvez répertorier les définitions de tâches de votre compte à tout moment à l'aide de la commande list-task-definitions. La sortie de cette commande affiche les valeurs `family` et `revision` que vous pouvez utiliser ensemble lorsque vous appelez create-service.

```
aws ecs list-task-definitions
```

Sortie :

```
{
    "taskDefinitionArns": [
```

```
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/sleep360:2",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/nginx-task:1",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:3",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:4",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:5",
        "arn:aws:ec2:us-east-1:aws_account_id:task-definition/wordpress:6"
    ]
}
```

Créer un service

Après avoir enregistré une tâche pour votre compte et lancé une instance de conteneur enregistrée dans votre cluster, vous pouvez créer un service Amazon ECS qui exécute et gère le nombre souhaité de tâches simultanément à l'aide de la définition de tâche que vous avez enregistrée. Dans cet exemple, vous placez une seule instance de la définition de `nginx:1` tâche dans votre MyCluster cluster.

```
aws ecs create-service --cluster MyCluster --service-name nginx-service --task-
definition nginx-task:1 --desired-count 1
```

Sortie :

```
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
        "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/nginx-
task:1",
        "desiredCount": 1,
        "runningCount": 0,
        "pendingCount": 0,
        "launchType": "EC2",
        "status": "ACTIVE",
        "createdAt": "2025-01-13T10:45:00.000Z"
    }
}
```

Liste des services

Affichez les services de votre cluster. Vous devriez voir le service que vous avez exécuté dans la section précédente. Vous pouvez noter l'ID du service ou l'ARN complet qui est renvoyé par cette commande et l'utiliser pour décrire le service ultérieurement.

```
aws ecs list-services --cluster MyCluster
```

Sortie :

```
{
    "taskArns": [
        "arn:aws:ecs:us-east-1:aws_account_id:task/task_ID"
    ]
}
```

Description des services

Décrivez le service à l'aide de la commande suivante pour obtenir plus d'informations sur le service.

```
aws ecs describe-services --cluster MyCluster --services nginx-service
```

Sortie :

```
{
    "services": [
        {
            "serviceArn": "arn:aws:ecs:us-east-1:aws_account_id:service/MyCluster/
nginx-service",
            "serviceName": "nginx-service",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinition": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "desiredCount": 1,
            "runningCount": 1,
            "pendingCount": 0,
            "launchType": "EC2",
            "status": "ACTIVE",
            "createdAt": "2025-01-13T10:45:00.000Z",
            "events": [
                {
```

```
                    "id": "abcd1234-5678-90ab-cdef-1234567890ab",
                    "createdAt": "2025-01-13T10:45:30.000Z",
                    "message": "(service nginx-service) has started 1 tasks: (task
 abcd1234-5678-90ab-cdef-1234567890ab)."
                }
            ]
        }
    ]
}
```

Description de la tâche en cours d'exécution

Après avoir décrit le service, exécutez la commande suivante pour obtenir plus d'informations sur la tâche en cours d'exécution dans le cadre de votre service.

```
aws ecs list-tasks --cluster MyCluster --service-name nginx-service
```

Sortie :

```
{
    "tasks": [
        {
            "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
            "clusterArn": "arn:aws:ecs:us-east-1:aws_account_id:cluster/MyCluster",
            "taskDefinitionArn": "arn:aws:ecs:us-east-1:aws_account_id:task-definition/
nginx-task:1",
            "containerInstanceArn": "arn:aws:ecs:us-east-1:aws_account_id:container-
instance/MyCluster/abcd1234-5678-90ab-cdef-1234567890ab",
            "lastStatus": "RUNNING",
            "desiredStatus": "RUNNING",
            "containers": [
                {
                    "containerArn": "arn:aws:ecs:us-east-1:aws_account_id:container/
MyCluster/abcd1234-5678-90ab-cdef-1234567890ab/abcd1234-5678-90ab-cdef-1234567890ab",
                    "taskArn": "arn:aws:ecs:us-east-1:aws_account_id:task/MyCluster/
abcd1234-5678-90ab-cdef-1234567890ab",
                    "name": "nginx",
                    "lastStatus": "RUNNING",
                    "networkBindings": [
                        {
                            "bindIP": "0.0.0.0",
```

```
                            "containerPort": 80,
                            "hostPort": 80,
                            "protocol": "tcp"
                        }
                    ]
                }
            ],
            "createdAt": "2025-01-13T10:45:00.000Z",
            "startedAt": "2025-01-13T10:45:30.000Z"
        }
    ]
}
```

Test du service Web

Pour tester le serveur Web

1. Récupérez l'adresse IP publique de votre instance de conteneur en exécutant la commande suivante.

   ```
   aws ec2 describe-instances --instance-ids i-abcd1234 --query
    'Reservations[0].Instances[0].PublicIpAddress' --output text
   ```

   Sortie :

   ```
   203.0.113.25
   ```

2. Après avoir récupéré l'adresse IP, exécutez la commande `curl` suivante avec l'adresse IP.

   ```
   curl http://203.0.113.25
   ```

   Sortie :

   ```
   <!DOCTYPE html>
   <html>
   <head>
   <title>Welcome to nginx!</title>
   ...
   </head>
   <body>
   <h1>Welcome to nginx!</h1>
   ```

```
<p>If you can see this page, the nginx web server is successfully installed and
 working.</p>
...
</body>
</html>
```

La page d'accueil de nginx confirme que votre service fonctionne correctement et est accessible depuis Internet.

nettoyer des ressources ;

Pour éviter des frais supplémentaires, supprimez les ressources que vous avez créées dans ce didacticiel.

Pour nettoyer des ressources

1. Mettez à jour le service pour qu'il n'y ait aucune tâche souhaitée, puis supprimez-le.

```
aws ecs update-service --cluster MyCluster --service nginx-service --desired-count
 0
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "desiredCount": 0,
        "runningCount": 1,
        "pendingCount": 0,
        "status": "ACTIVE"
    }
}
```

2. Attendez que les tâches en cours d'exécution s'arrêtent, puis supprimez le service.

```
aws ecs delete-service --cluster MyCluster --service nginx-service
{
    "service": {
        "serviceArn": "arn:aws:ecs:us-east-1:123456789012:service/MyCluster/nginx-
service",
        "serviceName": "nginx-service",
        "status": "DRAINING"
```

```
    }
}
```

3. Résiliez l'instance de conteneur que vous avez créée.

```
aws ec2 terminate-instances --instance-ids i-abcd1234
{
    "TerminatingInstances": [
        {
            "InstanceId": "i-abcd1234",
            "CurrentState": {
                "Code": 32,
                "Name": "shutting-down"
            },
            "PreviousState": {
                "Code": 16,
                "Name": "running"
            }
        }
    ]
}
```

4. Nettoyez le groupe de sécurité et la paire de clés que vous avez créés.

```
aws ec2 delete-security-group --group-id sg-abcd1234
aws ec2 delete-key-pair --key-name ecs-tutorial-key
rm ecs-tutorial-key.pem
```

5. Supprimez le cluster Amazon ECS.

```
aws ecs delete-cluster --cluster MyCluster
{
    "cluster": {
        "clusterArn": "arn:aws:ecs:us-east-1:123456789012:cluster/MyCluster",
        "clusterName": "MyCluster",
        "status": "INACTIVE"
    }
}
```

Configuration d'Amazon ECS pour écouter CloudWatch les événements

Découvrez comment configurer une fonction Lambda simple qui écoute les événements liés aux tâches et les enregistre dans un flux de CloudWatch log Logs.

Prérequis : Configuration d'un cluster test

Si vous ne possédez pas de cluster en cours d'exécution à partir duquel capturer des événements, suivez les étapes dans the section called “Création d’un cluster pour les charges de travail Fargate” pour en créer un. À la fin de ce didacticiel, vous exécutez une tâche sur ce cluster pour tester que la configuration de votre fonction Lambda est correcte.

Étape 1 : Créer la fonction Lambda

Au cours de cette procédure, vous allez créer une fonction Lambda simple à utiliser comme cible pour les messages de flux d'événements Amazon ECS.

1. Ouvrez la AWS Lambda console à l'adresse https://console.aws.amazon.com/lambda/.
2. Sélectionnez Create function (Créer une fonction).
3. Dans l'écran Author from scratch (Créer à partir de zéro), procédez comme suit :
   a. Pour Name (Nom), saisissez une valeur.
   b. Pour Runtime (Exécution), choisissez votre version de Python, par exemple, Python 3.9.
   c. Pour Role (Rôle), choisissez Create a new role with basic Lambda permissions (Créer un nouveau rôle avec les autorisations Lambda de base).
4. Sélectionnez Create function (Créer une fonction).
5. Dans la section Code de fonction, modifiez l'exemple de code selon l'exemple suivant :

```
import json

def lambda_handler(event, context):
    if event["source"] != "aws.ecs":
       raise ValueError("Function only supports input from events with a source
 type of: aws.ecs")

    print('Here is the event:')
```

```
    print(json.dumps(event))
```

Il s'agit d'une fonction Python 3.9 simple qui imprime les événements envoyés par Amazon ECS. Si tout est correctement configuré, à la fin de ce didacticiel, vous verrez que les détails de l'événement apparaissent dans le flux de log CloudWatch Logs associé à cette fonction Lambda.

6. Choisissez Enregistrer.

Étape 2 : Enregistrer une règle d'événement

Ensuite, vous créez une règle d' CloudWatch événements qui capture les événements de tâches provenant de vos clusters Amazon ECS. Cette règle capture tous les événements provenant de tous les clusters du compte dans lequel il est défini. Les messages de tâche eux-mêmes contiennent des informations sur la source de l'événement, y compris le cluster sur lequel elle réside, que vous pouvez utiliser pour filtrer et trier les événements par programmation.

> **Note**
>
> Lorsque vous utilisez la règle Console de gestion AWS pour créer un événement, la console ajoute automatiquement les autorisations IAM nécessaires pour autoriser CloudWatch Events à appeler votre fonction Lambda. Si vous créez une règle d'événement à l'aide du AWS CLI, vous devez accorder cette autorisation de manière explicite. Pour plus d'informations, consultez les sections Événements sur Amazon EventBridge et Modèles d' EventBridge événements Amazon dans le guide de EventBridge l'utilisateur Amazon.

Pour acheminer des événements vers votre fonction Lambda

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Events (Événements), Rules (Règles), Create rule (Créer une règle).
3. Pour Event Source (Source de l'événement), choisissez ECS comme source d'événement. Par défaut, la règle s'applique à tous les événements Amazon ECS pour tous vos groupes Amazon ECS. Sinon, vous pouvez sélectionner des événements spécifiques ou un groupe Amazon ECS spécifique.
4. Pour Targets (Cibles), choisissez Add target (Ajouter une cible), pour Target type (Type de cible), choisissez Lambda function (Fonction Lambda), puis sélectionnez votre fonction Lambda.

5. Choisissez Configure details (Configurer les détails).
6. Pour Rule definition (Définition de règle), saisissez un nom et une description pour la règle, puis choisissez Create rule (Créer une règle).

Étape 3 : Créer une définition de tâche

Créez une définition de tâche.

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, sélectionnez Task Definitions (Définition des tâches).
3. Choisissez Create new Task Definition (Créer une nouvelle définition de tâche), puis Create new revision with JSON (Créer une nouvelle révision avec JSON).
4. Copiez et collez l'exemple de définition de tâche suivant dans la zone, puis choisissez Save (Enregistrer).

```
{
   "containerDefinitions": [
      {
         "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\""
         ],
         "entryPoint": [
            "sh",
            "-c"
         ],
         "essential": true,
         "image": "public.ecr.aws/docker/library/httpd:2.4",
         "logConfiguration": {
            "logDriver": "awslogs",
            "options": {
               "awslogs-group" : "/ecs/fargate-task-definition",
               "awslogs-region": "us-east-1",
               "awslogs-stream-prefix": "ecs"
            }
         },
```

```
         "name": "sample-fargate-app",
         "portMappings": [
            {
               "containerPort": 80,
               "hostPort": 80,
               "protocol": "tcp"
            }
         ]
      }
   ],
   "cpu": "256",
   "executionRoleArn": "arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family": "fargate-task-definition",
   "memory": "512",
   "networkMode": "awsvpc",
   "runtimePlatform": {
        "operatingSystemFamily": "LINUX"
    },
   "requiresCompatibilities": [
       "FARGATE"
    ]
}
```

5. Choisissez Créer.

Étape 4 : Test de la règle

Enfin, vous créez une règle d' CloudWatch événements qui capture les événements de tâches provenant de vos clusters Amazon ECS. Cette règle capture tous les événements provenant de tous les clusters du compte dans lequel il est défini. Les messages de tâche eux-mêmes contiennent des informations sur la source de l'événement, y compris le cluster sur lequel elle réside, que vous pouvez utiliser pour filtrer et trier les événements par programmation.

Pour tester la règle

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Choisissez Task definitions (Définitions des tâches).
3. Choisissez console-sample-app-static, puis choisissez Deploy (Déployer), Run new task (Exécuter une nouvelle tâche).
4. Pour Cluster, choisissez par défaut, puis choisissez Deploy (Déployer).

5. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
6. Dans le volet de navigation, choisissez Logs et sélectionnez le groupe de journaux pour votre fonction Lambda (par exemple,/aws/lambda/*my-function*).
7. Sélectionnez un flux de journaux pour afficher les données d'événement.

Envoi d’alertes Amazon Simple Notification Service pour les événements de tâche Amazon ECS arrêtés

Configurez une règle d' EventBridge événement Amazon qui capture uniquement les événements de tâche lorsque la tâche a cessé de s'exécuter parce que l'un de ses conteneurs essentiels s'est arrêté. L'événement envoie uniquement les événements de tâche ayant une propriété spécifique `stoppedReason` à la rubrique Amazon SNS désignée.

Prérequis : Configuration d'un cluster test

Si vous ne possédez pas de cluster en cours d'exécution à partir duquel capturer des événements, suivez les étapes dans Mise en route avec la console à l'aide de conteneurs Linux sur AWS Fargate pour en créer un. À la fin de ce didacticiel, vous exécuterez une tâche sur ce cluster pour vérifier que vous avez correctement configuré votre rubrique et votre EventBridge règle Amazon SNS.

Prérequis : configurer les autorisations pour Amazon SNS

EventBridge Pour autoriser la publication sur une rubrique Amazon SNS, utilisez les commandes aws sns get-topic-attributes et aws sns set-topic-attributes.

Pour obtenir des informations sur l’ajout de l’autorisation, consultez Autorisations Amazon SNS dans le Guide du développeur Amazon Simple Notification Service.

Ajoutez les autorisations suivantes :

```
{
  "Sid": "PublishEventsToMyTopic",
  "Effect": "Allow",
  "Principal": {
     "Service": "events.amazonaws.com"
  },
  "Action": "sns: Publish",
  "Resource": "arn:aws:sns:region:account-id:TaskStoppedAlert",
```

```
}
```

Étape 1 : Créer une rubrique Amazon SNS et s'y abonner

Pour ce didacticiel, vous configurez une rubrique Amazon SNS à utiliser comme une cible de l'événement pour votre nouvelle règle d'événement.

Pour plus d'informations sur la façon de créer et de s'abonner à une rubrique Amazon SNS, consultez Démarrer avec Amazon SNS dans le Guide du développeur Amazon Simple Notification Service et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Valeur
Type	Standard
Nom	TaskStoppedAlert
Protocole	E-mail
Endpoint	Une adresse e-mail à laquelle vous avez actuellement accès

Étape 2 : Enregistrer une règle d'événement

Ensuite, vous enregistrez une règle d'événement qui capture uniquement les événements d'arrêt de la tâche pour les tâches avec des conteneurs arrêtés.

Pour plus d'informations sur la création et l'abonnement à une rubrique Amazon SNS, consultez la section Créer une règle EventBridge dans Amazon dans le guide de EventBridge l'utilisateur Amazon et utilisez le tableau suivant pour déterminer les options à sélectionner.

Option	Value
Type de règle	Règle avec un modèle d'événement

Option	Value
Source de l’événement	AWS événements ou événements EventBridge partenaires
Modèle d’événement	Modèle personnalisé (éditeur JSON)

Modèle d’événement	{
	"source":[
	"aws.ecs"
	],
	"detail-type":[
	"ECS Task State Change"
	],
	"detail":{
	"lastStatus":[
	"STOPPED"
	],
	"stoppedReason":[
	"Essential container in task exited"
	]
	}
	}

Type de cible	AWS service
Cible	Rubrique SNS
Rubrique	TaskStoppedAlert (Le sujet que vous avez créé à l'étape 1)

Étape 3 : Test de la règle

Vérifiez que la règle fonctionne en exécutant une tâche qui se ferme peu après son démarrage. Si votre règle d'événement est configurée correctement, vous recevez un message électronique en quelques minutes avec le texte de l'événement. Si vous disposez d'une définition de tâche existante qui peut satisfaire aux exigences de la règle, exécutez une tâche à l'aide de celle-ci. Si vous ne le faites pas, les étapes suivantes vous permettront d'enregistrer une définition de tâche Fargate et de l'exécuter.

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Dans le panneau de navigation, choisissez Task definitions (Définition des tâches).
3. Choisissez Create new task definition (Créer une nouvelle définition de tâche), puis Create new task definition with JSON (Créer une nouvelle définition de tâche avec JSON).
4. Dans la zone de l'éditeur JSON, modifiez votre fichier JSON, copiez ce qui suit dans l'éditeur.

```
{
   "containerDefinitions":[
      {
         "command":[
            "sh",
            "-c",
            "sleep 5"
         ],
         "essential":true,
         "image":"public.ecr.aws/amazonlinux/amazonlinux:latest",
         "name":"test-sleep"
      }
   ],
   "cpu":"256",
   "executionRoleArn":"arn:aws:iam::012345678910:role/ecsTaskExecutionRole",
   "family":"fargate-task-definition",
   "memory":"512",
   "networkMode":"awsvpc",
   "requiresCompatibilities":[
      "FARGATE"
   ]
}
```

5. Choisissez Créer.

Pour exécuter une tâche à partir de la console

1. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
2. Sur la page Clusters, choisissez le cluster que vous avez créé dans les conditions préalables.
3. Sous l'onglet Tasks (Tâches), choisissez Run new task (Exécuter une nouvelle tâche).
4. Pour Application type (Type d'application), choisissez Task (Tâche).
5. Pour Task definition (Définition de tâche), sélectionnez fargate-task-definition.
6. Pour Desired tasks (Tâches souhaitées), saisissez le nombre de tâches à lancer.
7. Choisissez Créer.

Concaténation des messages du journal Amazon ECS multiligne ou de suivi de pile

À partir de AWS la version 2.22.0 de Fluent Bit, un filtre multiligne est inclus. Le filtre multiligne aide à concaténer les messages de journaux qui appartiennent à l'origine à un seul contexte, mais qui ont été divisés en plusieurs enregistrements ou lignes de journal. Pour plus d'informations sur le filtre multiligne, consultez la documentation de Fluent Bit.

Voici des exemples courants de messages de journaux divisés :

- Suivis de pile.
- Applications qui impriment des journaux sur plusieurs lignes.
- Messages de journal qui ont été divisés, car ils étaient plus longs que la taille maximale de mémoire tampon d'exécution spécifiée. Vous pouvez concaténer les messages de journal divisés par le moteur d'exécution du conteneur en suivant l'exemple suivant GitHub : FireLens Exemple : Partial/Split Concaténer les journaux de conteneur.

Autorisations IAM requises

Vous disposez des autorisations IAM nécessaires pour que l'agent du conteneur extrait les images du conteneur depuis Amazon ECR et pour que le conteneur achemine les journaux vers CloudWatch Logs.

Pour ces autorisations, vous devez disposer des rôles suivants :

- Un rôle IAM de tâche.
- Un rôle IAM d’exécution de tâche

Vous avez besoin des autorisations suivantes :

- `logs:CreateLogStream`
- `logs:CreateLogGroup`
- `logs:PutLogEvents`

Déterminez à quel moment utiliser le paramètre de journal multiligne

Vous trouverez ci-dessous des exemples d'extraits de journal que vous pouvez voir dans la console CloudWatch Logs avec le paramètre de journal par défaut. Vous pouvez regarder la ligne qui commence par `log` pour déterminer si vous avez besoin du filtre multiligne. Lorsque le contexte est le même, vous pouvez utiliser le paramètre de journal multiligne. Dans cet exemple, le contexte est « com.myproject.model ». MyProject».

```
2022-09-20T15:47:56:595-05-00                            {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-
app", "source=": "stdout", "log": ": "     at com.myproject.modele.
(MyProject.badMethod.java:22)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log": ": "     at com.myproject.model.MyProject.badMethod(MyProject.java:22)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

```
2022-09-20T15:47:56:595-05-00                            {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app", "stdout",
 "log": ": "     at com.myproject.modele.(MyProject.oneMoreMethod.java:18)",
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
```

```
      "source": "stdout",
      "log": ": "     at
 com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE,
      "ecs_task_definition": "firelense-example-multiline:3"
     }
```

Une fois que vous avez utilisé le paramètre de journal multiligne, le résultat ressemblera à celui de l'exemple.

```
2022-09-20T15:47:56:595-05-00                              {"container_id":
 "82ba37cada1d44d389b03e78caf74faa-EXAMPLE", "container_name": "example-app",
 "stdout",...
    {
      "container_id":  "82ba37cada1d44d389b03e78caf74faa-EXAMPLE",
      "container_name: ": "example-app",
      "source": "stdout",
      "log:    "September 20, 2022 06:41:48 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
    at com.myproject.model.MyProject.oneMoreMethod(MyProject.java:18)
 com.myproject.module.MyProject.main(MyProject.java:6)",
      "ecs_cluster": "default",
      "ecs_task_arn": "arn:aws:region:123456789012:task/default/
b23c940d29ed4714971cba72cEXAMPLE",
      "ecs_task_definition": "firelense-example-multiline:2"
     }
```

Options d'analyse et de concaténation

Pour analyser les journaux et concaténer des lignes qui ont été divisées en raison de sauts de lignes, vous pouvez utiliser l'une de ces deux options.

- Utiliser votre propre fichier d'analyseur qui contient les règles pour analyser et concaténer les lignes qui appartiennent au même message.
- Utiliser un analyseur intégré de Fluent Bit. Pour une liste des langues supportées par les analyseurs intégrés de Fluent Bit, consultez la documentation de Fluent Bit.

Le didacticiel suivant vous guide à travers les étapes pour chaque cas d'utilisation. Les étapes vous montrent comment concaténer des lignes multiples et envoyer les journaux à Amazon. CloudWatch Vous pouvez spécifier une destination différente pour vos journaux.

Exemple : Utiliser un analyseur que vous créez

Dans cet exemple, vous allez réaliser les étapes suivantes :

1. Créer et charger l'image d'un conteneur Fluent Bit.
2. Créer et charger l'image d'une application multiligne de démonstration qui s'exécute, échoue, et génère un suivi de pile multiligne.
3. Créer la définition de tâche et exécuter la tâche.
4. Afficher les journaux pour vérifier que les messages qui couvrent plusieurs lignes apparaissent concaténés.

Création et chargement de l'image d'un conteneur Fluent Bit

Cette image inclura le fichier d'analyseur où vous spécifiez l'expression régulière et un fichier de configuration qui fait référence au fichier d'analyseur.

1. Créez un dossier avec le nom `FluentBitDockerImage`.
2. Dans ce dossier, créez un fichier d'analyseur qui contient les règles pour analyser le journal et concaténer les lignes qui appartiennent au même message.

   a. Collez le contenu suivant dans le fichier d'analyseur :

```
[MULTILINE_PARSER]
    name          multiline-regex-test
    type          regex
    flush_timeout 1000
    #
    # Regex rules for multiline parsing
    # ---------------------------------
    #
    # configuration hints:
    #
    #  - first state always has the name: start_state
    #  - every field in the rule must be inside double quotes
    #
    # rules |   state name  | regex pattern                  | next state
```

```
    # ------|---------------|--------------------------------------------
    rule      "start_state"   "/(Dec \d+ \d+\:\d+\:\d+)(.*)/"  "cont"
    rule      "cont"          "/^\s+at.*/"                     "cont"
```

Lorsque vous personnalisez votre modèle d'expression régulière, nous vous recommandons d'utiliser un éditeur d'expressions régulières pour tester l'expression.

b. Enregistrez le fichier sous le nom `parsers_multiline.conf`.

3. Dans le dossier `FluentBitDockerImage`, créez un fichier de configuration personnalisé qui fait référence au fichier d'analyseur que vous avez créé à l'étape précédente.

   Pour plus d'informations sur le fichier de configuration personnalisé, consultez Spécification d'un fichier de configuration personnalisé dans le Guide du développeur Amazon Elastic Container Service

   a. Collez le contenu suivant dans le fichier :

```
[SERVICE]
    flush                 1
    log_level             info
    parsers_file          /parsers_multiline.conf

[FILTER]
    name                  multiline
    match                 *
    multiline.key_content log
    multiline.parser      multiline-regex-test
```

> **Note**
>
> Vous devez utiliser le chemin absolu de l'analyseur.

   b. Enregistrez le fichier sous le nom `extra.conf`.

4. Dans le dossier `FluentBitDockerImage`, créez le Dockerfile avec l'image Fluent Bit, l'analyseur et les fichiers de configuration que vous avez créés.

   a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
```

```
ADD parsers_multiline.conf /parsers_multiline.conf
ADD extra.conf /extra.conf
```

b. Enregistrez le fichier sous le nom `Dockerfile`.

5. En utilisant le Dockerfile, créez une image Fluent Bit personnalisée avec l'analyseur et les fichiers de configuration personnalisés inclus.

> **Note**
>
> Vous pouvez placer le fichier d'analyseur et le fichier de configuration n'importe où dans l'image Docker, sauf lorsque `/fluent-bit/etc/fluent-bit.conf` ce chemin de fichier est utilisé par. FireLens

a. Créez l'image : `docker build -t fluent-bit-multiline-image.`

Où : `fluent-bit-multiline-image` est le nom de l'image dans cet exemple.

b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=fluent-bit-multiline-image`

En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image personnalisée Fluent Bit dans Amazon Elastic Container Registry.

a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

Où : `fluent-bit-multiline-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

La sortie vous donne les détails du nouveau référentiel.

b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`fluent-bit-multiline-image repositoryUri`*

Exemple : `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

Dans la sortie, le nom du référentiel passe de fluent-bit-multiline-repo à `repositoryUri`.

d. Authentifiez-vous auprès d'Amazon ECR en exécutant la commande `aws ecr get-login-password` et en spécifiant l'ID de registre auquel vous voulez vous authentifier : `aws ecr get-login-password | docker login --username AWS --password-stdin registry ID.dkr.ecr.region.amazonaws.com`

Exemple : `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

Un message de connexion réussie apparaît.

e. Envoyez (push) l'image vers Amazon ECR : `docker push registry ID.dkr.ecr.region.amazonaws.com/repository name`

Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Création et chargement de l'image pour une application multiligne de démonstration

Cette image comprendra un fichier script Python qui exécute l'application et un exemple de fichier journal.

Lorsque vous exécutez la tâche, l'application simule l'exécution, puis échoue et crée un suivi de pile.

1. Créez un dossier nommé `multiline-app` : `mkdir multiline-app`
2. Créez un fichier script Python.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `main.py`.

   b. Collez le contenu suivant dans le fichier :

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
    time.sleep(1)
```

```
# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Enregistrez le fichier `main.py`.

3. Créez un fichier journal d'exemple.

a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `test.log`.

b. Collez le contenu suivant dans le fichier :

```
single line...
Dec 14 06:41:08 Exception in thread "main" java.lang.RuntimeException:
 Something has gone wrong, aborting!
    at com.myproject.module.MyProject.badMethod(MyProject.java:22)
    at com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)
    at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)
    at com.myproject.module.MyProject.someMethod(MyProject.java:10)
    at com.myproject.module.MyProject.main(MyProject.java:6)
another line...
```

c. Enregistrez le fichier `test.log`.

4. Dans le dossier `multiline-app`, créez le Dockerfile.

a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/amazonlinux/amazonlinux:latest
ADD test.log /test.log

RUN yum upgrade -y && yum install -y python3

WORKDIR /usr/local/bin

COPY main.py .

CMD ["python3", "main.py"]
```

b. Enregistrez le fichier `Dockerfile`.

5. À l'aide du Dockerfile, créez une image.

   a. Créez l'image : `docker build -t multiline-app-image`

      Où : `multiline-app-image` est le nom de l'image dans cet exemple.

   b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=multiline-app-image`

      En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Où : `multiline-app-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel. Notez la valeur de `repositoryUri`, car vous en aurez besoin dans les étapes suivantes.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`multiline-app-image repositoryUri`*

      Exemple : `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

   c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

      Dans la sortie, le nom du référentiel passe de `multiline-app-repo` à la valeur de `repositoryUri`.

   d. Envoyez (push) l'image vers Amazon ECR : `docker push` *`aws_account_id`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

      Exemple : `docker push` *`xxxxxxxxxxxx`*`.dkr.ecr.`*`us-east-1`*`.amazonaws.com/multiline-app-repo`

Création de la définition de tâche et exécution de la tâche

1. Créez un fichier de définition de tâche avec le nom de fichier `multiline-task-definition.json`.
2. Collez le contenu suivant dans le fichier `multiline-task-definition.json` :

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN,
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
```

```
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Remplacez les éléments suivants dans la définition de tâche `multiline-task-definition.json` :

a. *`task role ARN`*

Pour trouver l'ARN du rôle de la tâche, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle de tâche `ecs-task-role-for-firelens` que vous avez créé. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

b. *`execution role ARN`*

Pour trouver l'ARN du rôle d'exécution, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle `ecsTaskExecutionRole`. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

c. *`aws_account_id`*

Pour trouver votre `aws_account_id`, connectez-vous à la Console de gestion AWS. Choisissez votre nom d'utilisateur et copiez votre identifiant de compte.

d. *`us-east-1`*

Remplacez la région si nécessaire.

3. Enregistrez le fichier de définition de tâche : `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region` *`region`*
4. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
5. Dans le volet de navigation, choisissez Définitions de tâches, puis choisissez la `firelens-example-multiline` famille, car nous avons enregistré la définition de tâche dans cette famille dans la première ligne de la définition de tâche précédente.
6. Choisissez la dernière version.
7. Choisissez Déployer, Exécuter la tâche.
8. Sur la page Exécuter la tâche, pour Cluster, choisissez le cluster, puis sous Mise en réseau, pour Sous-réseaux, choisissez les sous-réseaux disponibles pour votre tâche.
9. Choisissez Créer.

Vérifiez que les messages de journal multilignes dans Amazon CloudWatch apparaissent concaténés

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, développez Logs (Journaux) et choisissez Log groups (Groupes de journaux).
3. Choisissez le groupe de journaux `multiline-test/applicatio`.
4. Choisissez le journal. Affichez les messages. Les lignes qui correspondent aux règles du fichier analyseur sont concaténées et apparaissent comme un seul message.

   L'extrait de journal suivant montre les lignes concaténées dans un seul événement de suivi de pile Java :

```
{
    "container_id": "xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!\n
 at com.myproject.module.MyProject.badMethod(MyProject.java:22)\n    at
 com.myproject.module.MyProject.oneMoreMethod(MyProject.java:18)\n
 at com.myproject.module.MyProject.anotherMethod(MyProject.java:14)\n
 at com.myproject.module.MyProject.someMethod(MyProject.java:10)\n    at
 com.myproject.module.MyProject.main(MyProject.java:6)",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

   L'extrait de journal suivant montre comment le même message apparaît avec une seule ligne si vous exécutez un conteneur Amazon ECS qui n'est pas configuré pour concaténer les messages de journaux multilignes.

```
{
    "log": "Dec 14 06:41:08 Exception in thread \"main\"
 java.lang.RuntimeException: Something has gone wrong, aborting!",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
```

```
    "ecs_task_definition": "firelens-example-multiline:3"
}
```

Exemple : Utilisation d'un analyseur intégré de Fluent Bit

Dans cet exemple, vous allez réaliser les étapes suivantes :

1. Créer et charger l'image d'un conteneur Fluent Bit.
2. Créer et charger l'image d'une application multiligne de démonstration qui s'exécute, échoue, et génère un suivi de pile multiligne.
3. Créer la définition de tâche et exécuter la tâche.
4. Afficher les journaux pour vérifier que les messages qui couvrent plusieurs lignes apparaissent concaténés.

Création et chargement de l'image d'un conteneur Fluent Bit

Cette image comprendra un fichier de configuration qui fait référence à l'analyseur de Fluent Bit.

1. Créez un dossier avec le nom `FluentBitDockerImage`.
2. Dans le dossier `FluentBitDockerImage`, créez un fichier de configuration personnalisé qui fait référence au fichier d'analyseur intégré de Fluent Bit.

   Pour plus d'informations sur le fichier de configuration personnalisé, consultez Spécification d'un fichier de configuration personnalisé dans le Guide du développeur Amazon Elastic Container Service

   a. Collez le contenu suivant dans le fichier :

   ```
   [FILTER]
       name                  multiline
       match                 *
       multiline.key_content log
       multiline.parser      go
   ```

   b. Enregistrez le fichier sous le nom `extra.conf`.
3. Dans le dossier `FluentBitDockerImage`, créez le Dockerfile avec l'image Fluent Bit, l'analyseur et les fichiers de configuration que vous avez créés.

a. Collez le contenu suivant dans le fichier :

```
FROM public.ecr.aws/aws-observability/aws-for-fluent-bit:latest
ADD extra.conf /extra.conf
```

b. Enregistrez le fichier sous le nom `Dockerfile`.

4. En utilisant le Dockerfile, créez une image Fluent Bit personnalisée avec le fichier de configuration personnalisé inclus.

> **Note**
>
> Vous pouvez placer le fichier de configuration n'importe où dans l'image Docker, sauf si `/fluent-bit/etc/fluent-bit.conf` ce chemin de fichier est utilisé par FireLens.

a. Créez l'image : `docker build -t fluent-bit-multiline-image.`

Où : `fluent-bit-multiline-image` est le nom de l'image dans cet exemple.

b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=fluent-bit-multiline-image`

En cas de succès, la sortie montre l'image et l'identification `latest`.

5. Chargez l'image personnalisée Fluent Bit dans Amazon Elastic Container Registry.

a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name fluent-bit-multiline-repo --region us-east-1`

Où : `fluent-bit-multiline-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

La sortie vous donne les détails du nouveau référentiel.

b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`fluent-bit-multiline-image repositoryUri`*

Exemple : `docker tag fluent-bit-multiline-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=`*`repositoryUri`*

   Dans la sortie, le nom du référentiel passe de fluent-bit-multiline-repo à `repositoryUri`.

d. Authentifiez-vous auprès d'Amazon ECR en exécutant la commande `aws ecr get-login-password` et en spécifiant l'ID de registre auquel vous voulez vous authentifier : `aws ecr get-login-password | docker login --username AWS --password-stdin` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com`

   Exemple : `ecr get-login-password | docker login --username AWS --password-stdin xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com`

   Un message de connexion réussie apparaît.

e. Envoyez (push) l'image vers Amazon ECR : `docker push` *`registry ID`*`.dkr.ecr.`*`region`*`.amazonaws.com/`*`repository name`*

   Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-multiline-repo`

Création et chargement de l'image pour une application multiligne de démonstration

Cette image comprendra un fichier script Python qui exécute l'application et un exemple de fichier journal.

1. Créez un dossier nommé `multiline-app` : `mkdir multiline-app`
2. Créez un fichier script Python.

   a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `main.py`.

   b. Collez le contenu suivant dans le fichier :

```
import os
import time
file1 = open('/test.log', 'r')
Lines = file1.readlines()

count = 0

for i in range(10):
    print("app running normally...")
```

```
    time.sleep(1)

# Strips the newline character
for line in Lines:
    count += 1
    print(line.rstrip())
print(count)
print("app terminated.")
```

c. Enregistrez le fichier `main.py`.

3. Créez un fichier journal d'exemple.

a. Dans le dossier `multiline-app`, créez un fichier et nommez-le `test.log`.

b. Collez le contenu suivant dans le fichier :

```
panic: my panic

goroutine 4 [running]:
panic(0x45cb40, 0x47ad70)
  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8 sp=0xc42003f710
 pc=0x422f7c
main.main.func1(0xc420024120)
  foo.go:6 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0
 sp=0xc42003f7d8 pc=0x44b4d1
created by main.main
  foo.go:5 +0x58

goroutine 1 [chan receive]:
runtime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00
 pc=0x42503c
runtime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e
runtime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4
runtime.chanrecv1(0xc420024120, 0x0)
  /usr/local/go/src/runtime/chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20
 pc=0x40439b
```

```
main.main()
  foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef
runtime.main()
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc420053fe8
 sp=0xc420053fe0 pc=0x44b4d1

goroutine 2 [force gc (idle)]:
runtime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c
runtime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e
runtime.forcegchelper()
  /usr/local/go/src/runtime/proc.go:238 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8
 pc=0x424e5c
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8
 sp=0xc42003e7e0 pc=0x44b4d1
created by runtime.init.4
  /usr/local/go/src/runtime/proc.go:227 +0x35

goroutine 3 [GC sweep wait]:
runtime.gopark(0x4739b8, 0x4ad7e0, 0x46fdd2, 0xd, 0x419914, 0x1)
  /usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003ef60 sp=0xc42003ef30
 pc=0x42503c
runtime.goparkunlock(0x4ad7e0, 0x46fdd2, 0xd, 0x14, 0x1)
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003efa0 sp=0xc42003ef60
 pc=0x42512e
runtime.bgsweep(0xc42001e150)
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8
 sp=0xc42003efa0 pc=0x419973
runtime.goexit()
  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003efe0
 sp=0xc42003efd8 pc=0x44b4d1
created by runtime.gcenable
  /usr/local/go/src/runtime/mgc.go:216 +0x58
one more line, no multiline
```

c. Enregistrez le fichier `test.log`.

4. Dans le dossier `multiline-app`, créez le Dockerfile.

   a. Collez le contenu suivant dans le fichier :

   ```
   FROM public.ecr.aws/amazonlinux/amazonlinux:latest
   ADD test.log /test.log

   RUN yum upgrade -y && yum install -y python3

   WORKDIR /usr/local/bin

   COPY main.py .

   CMD ["python3", "main.py"]
   ```

   b. Enregistrez le fichier `Dockerfile`.

5. À l'aide du Dockerfile, créez une image.

   a. Créez l'image : `docker build -t multiline-app-image`

      Où : `multiline-app-image` est le nom de l'image dans cet exemple.

   b. Vérifiez que l'image a été créée correctement : `docker images –filter reference=multiline-app-image`

      En cas de succès, la sortie montre l'image et l'identification `latest`.

6. Chargez l'image dans Amazon Elastic Container Registry.

   a. Créez un référentiel Amazon ECR pour stocker l'image : `aws ecr create-repository --repository-name multiline-app-repo --region us-east-1`

      Où : `multiline-app-repo` est le nom du référentiel et `us-east-1` est la région dans cet exemple.

      La sortie vous donne les détails du nouveau référentiel. Notez la valeur de `repositoryUri`, car vous en aurez besoin dans les étapes suivantes.

   b. Étiquetez votre image avec la valeur `repositoryUri` de la sortie précédente : `docker tag` *`multiline-app-image repositoryUri`*

      Exemple : `docker tag multiline-app-image xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

c. Exécutez l'image docker pour vérifier qu'elle s'est exécutée correctement :`docker images –filter reference=repositoryUri`

   Dans la sortie, le nom du référentiel passe de `multiline-app-repo` à la valeur de `repositoryUri`.

d. Envoyez (push) l'image vers Amazon ECR : `docker push aws_account_id.dkr.ecr.region.amazonaws.com/repository name`

   Exemple : `docker push xxxxxxxxxxxx.dkr.ecr.us-east-1.amazonaws.com/multiline-app-repo`

Création de la définition de tâche et exécution de la tâche

1. Créez un fichier de définition de tâche avec le nom de fichier `multiline-task-definition.json`.
2. Collez le contenu suivant dans le fichier `multiline-task-definition.json` :

```
{
    "family": "firelens-example-multiline",
    "taskRoleArn": "task role ARN",
    "executionRoleArn": "execution role ARN",
    "containerDefinitions": [
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/fluent-bit-
multiline-image:latest",
            "name": "log_router",
            "firelensConfiguration": {
                "type": "fluentbit",
                "options": {
                    "config-file-type": "file",
                    "config-file-value": "/extra.conf"
                }
            },
            "memoryReservation": 50
        },
        {
            "essential": true,
            "image": "aws_account_id.dkr.ecr.us-east-1.amazonaws.com/multiline-app-
image:latest",
```

```
            "name": "app",
            "logConfiguration": {
                "logDriver": "awsfirelens",
                "options": {
                    "Name": "cloudwatch_logs",
                    "region": "us-east-1",
                    "log_group_name": "multiline-test/application",
                    "auto_create_group": "true",
                    "log_stream_prefix": "multiline-"
                }
            },
            "memoryReservation": 100
        }
    ],
    "requiresCompatibilities": ["FARGATE"],
    "networkMode": "awsvpc",
    "cpu": "256",
    "memory": "512"
}
```

Remplacez les éléments suivants dans la définition de tâche `multiline-task-definition.json` :

a. *task role ARN*

Pour trouver l'ARN du rôle de la tâche, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle de tâche `ecs-task-role-for-firelens` que vous avez créé. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

b. *execution role ARN*

Pour trouver l'ARN du rôle d'exécution, allez dans la console IAM. Choisissez Roles (Rôles) et trouvez le rôle `ecsTaskExecutionRole`. Choisissez le rôle et copiez l'ARN qui apparaît dans la section Summary (Résumé).

c. *aws_account_id*

Pour trouver votre `aws_account_id`, connectez-vous à la Console de gestion AWS. Choisissez votre nom d'utilisateur et copiez votre identifiant de compte.

d. *us-east-1*

Remplacez la région si nécessaire.

3. Enregistrez le fichier de définition de tâche : `aws ecs register-task-definition --cli-input-json file://multiline-task-definition.json --region us-east-1`
4. Ouvrez la console à l'adresse https://console.aws.amazon.com/ecs/v2.
5. Dans le volet de navigation, choisissez Définitions de tâches, puis choisissez la `firelens-example-multiline` famille, car nous avons enregistré la définition de tâche dans cette famille dans la première ligne de la définition de tâche précédente.
6. Choisissez la dernière version.
7. Choisissez Déployer, Exécuter la tâche.
8. Sur la page Exécuter la tâche, pour Cluster, choisissez le cluster, puis sous Mise en réseau, pour Sous-réseaux, choisissez les sous-réseaux disponibles pour votre tâche.
9. Choisissez Créer.

Vérifiez que les messages de journal multilignes dans Amazon CloudWatch apparaissent concaténés

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, développez Logs (Journaux) et choisissez Log groups (Groupes de journaux).
3. Choisissez le groupe de journaux `multiline-test/applicatio`.
4. Choisissez le journal et affichez les messages. Les lignes qui correspondent aux règles du fichier analyseur sont concaténées et apparaissent comme un seul message.

   L'extrait de journal suivant montre un suivi de pile Go qui est concaténé en un seul événement :

```
{
    "log": "panic: my panic\n\ngoroutine 4 [running]:\npanic(0x45cb40,
 0x47ad70)\n  /usr/local/go/src/runtime/panic.go:542 +0x46c fp=0xc42003f7b8
 sp=0xc42003f710 pc=0x422f7c\nmain.main.func1(0xc420024120)\n  foo.go:6
 +0x39 fp=0xc42003f7d8 sp=0xc42003f7b8 pc=0x451339\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003f7e0 sp=0xc42003f7d8
 pc=0x44b4d1\ncreated by main.main\n  foo.go:5 +0x58\n\ngoroutine 1 [chan receive]:
\nruntime.gopark(0x4739b8, 0xc420024178, 0x46fcd7, 0xc, 0xc420028e17, 0x3)\n  /usr/
local/go/src/runtime/proc.go:280 +0x12c fp=0xc420053e30 sp=0xc420053e00 pc=0x42503c
\nruntime.goparkunlock(0xc420024178, 0x46fcd7, 0xc, 0x1000f010040c217, 0x3)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc420053e70 sp=0xc420053e30
 pc=0x42512e\nruntime.chanrecv(0xc420024120, 0x0, 0xc420053f01, 0x4512d8)\n
  /usr/local/go/src/runtime/chan.go:506 +0x304 fp=0xc420053f20 sp=0xc420053e70
 pc=0x4046b4\nruntime.chanrecv1(0xc420024120, 0x0)\n  /usr/local/go/src/runtime/
```

```
chan.go:388 +0x2b fp=0xc420053f50 sp=0xc420053f20 pc=0x40439b\nmain.main()\n
 foo.go:9 +0x6f fp=0xc420053f80 sp=0xc420053f50 pc=0x4512ef\nruntime.main()\n
  /usr/local/go/src/runtime/proc.go:185 +0x20d fp=0xc420053fe0 sp=0xc420053f80
 pc=0x424bad\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337
 +0x1 fp=0xc420053fe8 sp=0xc420053fe0 pc=0x44b4d1\n\ngoroutine 2 [force gc
 (idle)]:\nruntime.gopark(0x4739b8, 0x4ad720, 0x47001e, 0xf, 0x14, 0x1)\n  /
usr/local/go/src/runtime/proc.go:280 +0x12c fp=0xc42003e768 sp=0xc42003e738
 pc=0x42503c\nruntime.goparkunlock(0x4ad720, 0x47001e, 0xf, 0xc420000114, 0x1)\n
  /usr/local/go/src/runtime/proc.go:286 +0x5e fp=0xc42003e7a8 sp=0xc42003e768
 pc=0x42512e\nruntime.forcegchelper()\n  /usr/local/go/src/runtime/proc.go:238
 +0xcc fp=0xc42003e7e0 sp=0xc42003e7a8 pc=0x424e5c\nruntime.goexit()\n  /usr/
local/go/src/runtime/asm_amd64.s:2337 +0x1 fp=0xc42003e7e8 sp=0xc42003e7e0
 pc=0x44b4d1\ncreated by runtime.init.4\n  /usr/local/go/src/runtime/proc.go:227
 +0x35\n\ngoroutine 3 [GC sweep wait]:\nruntime.gopark(0x4739b8, 0x4ad7e0,
 0x46fdd2, 0xd, 0x419914, 0x1)\n  /usr/local/go/src/runtime/proc.go:280 +0x12c
 fp=0xc42003ef60 sp=0xc42003ef30 pc=0x42503c\nruntime.goparkunlock(0x4ad7e0,
 0x46fdd2, 0xd, 0x14, 0x1)\n  /usr/local/go/src/runtime/proc.go:286 +0x5e
 fp=0xc42003efa0 sp=0xc42003ef60 pc=0x42512e\nruntime.bgsweep(0xc42001e150)\n
  /usr/local/go/src/runtime/mgcsweep.go:52 +0xa3 fp=0xc42003efd8 sp=0xc42003efa0
 pc=0x419973\nruntime.goexit()\n  /usr/local/go/src/runtime/asm_amd64.s:2337 +0x1
 fp=0xc42003efe0 sp=0xc42003efd8 pc=0x44b4d1\ncreated by runtime.gcenable\n  /usr/
local/go/src/runtime/mgc.go:216 +0x58",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:2"
}
```

L'extrait de journal suivant montre comment le même événement apparaît si vous exécutez un conteneur ECS qui n'est pas configuré pour concaténer les messages de journaux multilignes. Le champ du journal contient une seule ligne.

```
{
    "log": "panic: my panic",
    "container_id": "xxxxxx-xxxxxx",
    "container_name": "app",
    "source": "stdout",
    "ecs_cluster": "default",
    "ecs_task_arn": "arn:aws:ecs:us-east-1:xxxxxxxxxxxx:task/default/xxxxxx",
    "ecs_task_definition": "firelens-example-multiline:3"
```

> **Note**
>
> Si vos journaux sont envoyés dans des fichiers journaux au lieu de la sortie standard, nous vous recommandons de spécifier les paramètres de configuration `multiline.parser` et `multiline.key_content` dans le plugin d'entrée Tail au lieu du filtre.

Déploiement de Fluent Bit sur Amazon ECS pour des conteneurs Windows

Fluent Bit est un processeur et un routeur de journaux rapides et flexibles pris en charge par différents systèmes d'exploitation. Il peut être utilisé pour acheminer les journaux vers diverses AWS destinations telles qu'Amazon CloudWatch Logs, Firehose, Amazon S3 et Amazon OpenSearch Service. Fluent Bit prend en charge des solutions partenaires courantes, comme Datadog, Splunk et les serveurs HTTP personnalisés. Pour plus d'informations sur Fluent Bit, consultez le site web Fluent Bit .

L'image AWS pour Fluent Bit est disponible sur Amazon ECR à la fois sur la galerie publique Amazon ECR et dans un référentiel Amazon ECR dans la plupart des Régions pour une haute disponibilité. Pour plus d'informations, consultez aws-for-fluent-bit le GitHub site Web.

Ce didacticiel explique comment déployer des conteneurs Fluent Bit sur leurs instances Windows exécutées dans Amazon ECS afin de diffuser les journaux générés par les tâches Windows vers Amazon CloudWatch pour une journalisation centralisée.

Ce didacticiel utilise l'approche suivante :

- Fluent Bit fonctionne comme un service avec la stratégie de planification du démon. Cette stratégie garantit qu'une seule instance de Fluent Bit s'exécute toujours sur les instances de conteneur du cluster.
  - Écoute sur le port 24224 à l'aide du plug-in d'entrée directe.
  - Exposez le port 24224 à l'hôte afin que le moteur d'exécution Docker puisse envoyer des journaux à Fluent Bit en utilisant ce port exposé.
  - Dispose d'une configuration qui permet à Fluent Bit d'envoyer les enregistrements des journaux vers des destinations spécifiées.

- Lancement de tous les autres conteneurs de tâches Amazon ECS à l'aide du pilote de journalisation Fluentd. Pour plus d'informations, consultez Fluentd logging drive (Pilote de journalisation Fluentd) sur le site Web de la documentation de Docker.
  - Docker se connecte au socket TCP 24224 sur localhost dans l'espace de noms de l'hôte.
  - L'agent Amazon ECS ajoute des étiquettes aux conteneurs, notamment le nom du cluster, le nom de famille de la définition de la tâche, le numéro de révision de la définition de la tâche, l'ARN de la tâche et le nom du conteneur. Les mêmes informations sont ajoutées à l'enregistrement du journal à l'aide de l'option labels du pilote de journalisation fluentd de Docker. Pour de plus amples informations, veuillez consulter labels, labels-regex, env, and env-regex sur le site web de la documentation de Docker.
  - Comme l'option `async` du pilote de journalisation Fluentd est définie sur `true`, lorsque le conteneur Fluent Bit est redémarré, Docker met les journaux en mémoire tampon jusqu'à ce que le conteneur Fluent Bit soit redémarré. Vous pouvez augmenter la limite de mémoire tampon en définissant l'option fluentd-buffer-limit. Pour plus d'informations, consultez fluentd-buffer-limit sur le site Web de la documentation de Docker.

Le flux de travail est le suivant :

- Le conteneur Fluent Bit démarre et écoute sur le port 24224 qui est exposé à l'hôte.
- Fluent Bit utilise les informations d'identification du rôle IAM de la tâche spécifiées dans sa définition de tâche.
- Les autres tâches lancées sur la même instance utilisent le pilote de journalisation fluentd de Docker pour se connecter au conteneur Fluent Bit sur le port 24224.
- Lorsque les conteneurs d'applications génèrent des journaux, le moteur d'exécution de Docker étiquette ces enregistrements, ajoute des métadonnées supplémentaires spécifiées dans les étiquettes, puis les transmet sur le port 24224 de l'espace de noms de l'hôte.
- Fluent Bit reçoit l'enregistrement du journal sur le port 24224 car il est exposé à l'espace de noms de l'hôte.
- Fluent Bit effectue son traitement interne et achemine les journaux comme spécifié.

Ce didacticiel utilise la configuration CloudWatch Fluent Bit par défaut qui effectue les opérations suivantes :

- Crée un nouveau groupe de journaux pour chaque cluster et chaque famille de définitions de tâches.

- Crée un nouveau flux de journal pour chaque conteneur de tâches du groupe de journaux précédemment généré chaque fois qu'une nouvelle tâche est lancée. Chaque flux sera marqué avec l'identifiant de tâche à laquelle le conteneur appartient.
- Ajoute des métadonnées supplémentaires, notamment le nom du cluster, l'ARN de la tâche, le nom du conteneur de la tâche, la famille de définitions de la tâche et le numéro de révision de la définition de la tâche dans chaque entrée du journal.

  Par exemple, si vous avez `task_1` with `container_1` `container_2` et t `ask_2` with`container_3`, les flux de CloudWatch log sont les suivants :

  - `/aws/ecs/windows.ecs_task_1`

    `task-out.`*`TASK_ID`*`.container_1`

    `task-out.`*`TASK_ID`*`.container_2`
  - `/aws/ecs/windows.ecs_task_2`

    `task-out.`*`TASK_ID`*`.container_3`

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

Étapes

- Conditions préalables
- Étape 1 : Créer les rôles IAM d'accès
- Étape 2 : Créer une instance de conteneur Windows Amazon ECS
- Étape 3 : Configurer Fluent Bit
- Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch
- Étape 5 : Exécuter la définition de tâche ecs-windows-fluent-bit en tant que service Amazon ECS à l'aide de la stratégie de planification des démons
- Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux

- Étape 7 : Exécuter la définition de la tâche windows-app-task
- Étape 8 : Vérifiez les connexions CloudWatch
- Étape 9 : nettoyer

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations, consultez la section Installation ou mise à jour de la version la plus récente de l' AWS CLI.
- L'image du conteneur `aws-for-fluent-bit` est disponible pour les systèmes d'exploitation Windows suivants :
  - Windows Server 2019 Core
  - Windows Server 2019 Full
  - Windows Server 2022 Core
  - Windows Server 2022 Full
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Vous avez un cluster. Dans ce didacticiel, le nom du cluster est FluentBit-cluster.
- Vous disposez d'un VPC doté d'un sous-réseau public sur lequel l'instance EC2 sera lancée. Vous pouvez utiliser votre VPC par défaut. Vous pouvez également utiliser un sous-réseau privé qui permet aux CloudWatch points de terminaison Amazon d'accéder au sous-réseau. Pour plus d'informations sur les CloudWatch points de terminaison Amazon, consultez la section CloudWatch Points de terminaison et quotas Amazon dans le. Références générales AWS Pour plus d'informations sur la manière d'utiliser l'assistant Amazon VPC pour créer un VPC, consultez the section called “Créer un Virtual Private Cloud”.

Étape 1 : Créer les rôles IAM d'accès

Créez les rôles IAM Amazon ECS.

1. Créez le rôle d'instance de conteneur Amazon ECS nommé « ecs InstanceRole ». Pour plus d'informations, consultez Rôle IAM d'instance de conteneur Amazon ECS.
2. Créez un rôle IAM pour la tâche Fluent Bit intitulée `fluentTaskRole`. Pour de plus amples informations, veuillez consulter the section called “Rôle IAM de tâche”.

Les autorisations IAM accordées dans ce rôle IAM sont prises en charge par les conteneurs de tâches. Pour autoriser Fluent Bit à envoyer des journaux CloudWatch, vous devez associer les autorisations suivantes au rôle IAM de la tâche.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
    {
        "Effect": "Allow",
        "Action": [
            "logs:CreateLogStream",
            "logs:CreateLogGroup",
            "logs:DescribeLogStreams",
            "logs:PutLogEvents"
        ],
        "Resource": "*"
    }
    ]
}
```

3. Attachez la stratégie au rôle.

   a. Enregistrez le contenu précédent dans un fichier nommé`fluent-bit-policy.json`.

   b. Exécutez la commande suivante pour associer la politique en ligne au rôle IAM `fluentTaskRole`.

```
aws iam put-role-policy --role-name fluentTaskRole --policy-name
 fluentTaskPolicy --policy-document file://fluent-bit-policy.json
```

Étape 2 : Créer une instance de conteneur Windows Amazon ECS

Créez une instance de conteneur Windows Amazon ECS.

Pour créer une instance Amazon ECS

1. Utilisez la commande `aws ssm get-parameters` pour récupérer l'ID AMI de la région qui héberge votre VPC. Pour plus d'informations, consultez la section Extraction des métadonnées Amazon ECS-Optimized AMI.
2. Utilisez la console Amazon EC2 pour lancer l'instance.
   a. Ouvrez la console Amazon EC2 à l'adresse https://console.aws.amazon.com/ec2/.
   b. Dans la barre de navigation, sélectionnez la région à utiliser.
   c. Sur le tableau de bord EC2, sélectionnez Launch instance (Lancer une instance).
   d. Pour Name (Nom), saisissez un nom unique.
   e. Pour Application and OS Images (Amazon Machine Image) (Images de l'application et du SE (Amazon Machine Image)), choisissez l'AMI que vous avez récupérée lors de la première étape.
   f. Concernant l'option Instance type (Type d'instance), choisissez `t3.xlarge`.
   g. Pour Key pair (login) (Paire de clés (connexion)), choisissez une paire de clés.
   h. Sous Network Settings (Paramètres réseau), pour Security group (Groupe de sécurité), choisissez un groupe de sécurité existant ou créez-en un nouveau.
   i. Sous Paramètres réseau, pour IP Auto-assign publique, sélectionnez Activer.
   j. Sous Détails avancés, pour le profil d'instance IAM, sélectionnez ecs. InstanceRole
   k. Configurez votre instance de conteneur Amazon ECS avec les données utilisateur suivantes. Sous Détails avancés, collez le script suivant dans le champ Données utilisateur, en le *`cluster_name`* remplaçant par le nom de votre cluster.

```
<powershell>
Import-Module ECSTools
Initialize-ECSAgent -Cluster cluster-name -EnableTaskENI -EnableTaskIAMRole -
LoggingDrivers '["awslogs","fluentd"]'
</powershell>
```

   l. Lorsque vous êtes prêt, cochez la case de confirmation, puis sélectionnez Launch Instances (Lancer des instances).
   m. Une page de confirmation indique que l'instance est en cours de lancement. Sélectionnez View Instances (Afficher les instances) pour fermer la page de confirmation et revenir à la console.

Étape 3 : Configurer Fluent Bit

Vous pouvez utiliser la configuration par défaut suivante fournie par AWS pour démarrer rapidement :

- Amazon CloudWatch, qui est basé sur le plug-in Fluent Bit pour Amazon figurant dans CloudWatch le manuel officiel de Fluent Bit.

Vous pouvez également utiliser d'autres configurations par défaut fournies par AWS. Pour plus d'informations, consultez la section Overriding the entrypoint for the Windows image (Ignorer le point d'entrée de l'image Windows) sur le site Web `aws-for-fluent-bit` de Github.

La configuration par défaut d'Amazon CloudWatch Fluent Bit est la suivante.

Remplacez les variables suivantes :

- *`region`*avec la région dans laquelle vous souhaitez envoyer les CloudWatch journaux Amazon.

```
[SERVICE]
    Flush               5
    Log_Level           info
    Daemon              off

[INPUT]
    Name                forward
    Listen              0.0.0.0
    Port                24224
    Buffer_Chunk_Size   1M
    Buffer_Max_Size     6M
    Tag_Prefix          ecs.

# Amazon ECS agent adds the following log keys as labels to the docker container.
# We would use fluentd logging driver to add these to log record while sending it to
 Fluent Bit.
[FILTER]
    Name                modify
    Match               ecs.*
    Rename              com.amazonaws.ecs.cluster ecs_cluster
    Rename              com.amazonaws.ecs.container-name ecs_container_name
    Rename              com.amazonaws.ecs.task-arn ecs_task_arn
    Rename              com.amazonaws.ecs.task-definition-family
 ecs_task_definition_family
```

```
    Rename              com.amazonaws.ecs.task-definition-version
 ecs_task_definition_version

[FILTER]
    Name                rewrite_tag
    Match               ecs.*
    Rule                $ecs_task_arn ^([a-z-:0-9]+)/([a-zA-Z0-9-_]+)/([a-z0-9]+)$
 out.$3.$ecs_container_name false
    Emitter_Name        re_emitted

[OUTPUT]
    Name                cloudwatch_logs
    Match               out.*
    region              region
    log_group_name      fallback-group
    log_group_template  /aws/ecs/$ecs_cluster.$ecs_task_definition_family
    log_stream_prefix   task-
    auto_create_group   On
```

Chaque journal qui entre dans Fluent Bit possède une étiquette que vous spécifiez. Si vous n'en fournissez pas, elle est générée automatiquement. Les étiquettes peuvent être utilisées pour acheminer différents journaux vers différentes destinations. Pour plus d'informations, voir Tag (Étiquette) dans le manuel officiel de Fluent Bit.

La configuration Fluent Bit précédente possède les propriétés suivantes :

- Le plug-in d'entrée directe écoute le trafic entrant sur le port TCP 24224.
- Chaque entrée de journal reçue sur ce port possède une étiquette que le plug-in d'entrée directe modifie pour préfixer l'enregistrement avec une chaîne `ecs.`.
- Le pipeline interne de Fluent Bit achemine l'entrée du journal pour modifier le filtre à l'aide d'une correspondance d'expression régulière (regex). Ce filtre remplace les clés dans l'enregistrement de journal au format JSON par le format que Fluent Bit peut utiliser.
- L'entrée de journal modifiée est ensuite utilisée par le filtre rewrite_tag. Ce filtre remplace la balise de l'enregistrement du journal par le format out. *`TASK_ID`*. *`CONTAINER_NAME`*.
- La nouvelle balise sera acheminée vers le plug-in de sortie cloudwatch_logs qui crée les groupes de journaux et les flux comme décrit précédemment en utilisant les `log_stream_prefix` options `log_group_template` et du plug-in de sortie. CloudWatch Pour plus d'informations, voir Configuration parameters (Paramètres de configuration) dans le manuel officiel de Fluent Bit.

Étape 4 : enregistrer une définition de tâche Windows Fluent Bit qui achemine les journaux vers CloudWatch

Enregistrez une définition de tâche Windows Fluent Bit vers laquelle les journaux sont acheminés CloudWatch.

> **Note**
>
> Cette définition de tâche expose le port 24224 du conteneur Fluent Bit au port hôte 24224. Vérifiez que ce port n'est pas ouvert dans le groupe de sécurité de votre instance EC2 pour empêcher tout accès depuis l'extérieur.

Pour enregistrer une définition de tâche

1. Créez un fichier nommé `fluent-bit.json` avec les contenus suivants.

   Remplacez les variables suivantes :

   - *`task-iam-role`*avec le nom de ressource Amazon (ARN) de votre tâche (rôle IAM)
   - *`region`*avec la région dans laquelle s'exécute votre tâche

```
{
  "family": "ecs-windows-fluent-bit",
  "taskRoleArn": "task-iam-role",
  "containerDefinitions": [
    {
      "name": "fluent-bit",
      "image": "public.ecr.aws/aws-observability/aws-for-fluent-
bit:windowsservercore-latest",
      "cpu": 512,
      "portMappings": [
        {
          "hostPort": 24224,
          "containerPort": 24224,
          "protocol": "tcp"
        }
      ],
      "entryPoint": [
        "Powershell",
```

```
        "-Command"
      ],
      "command": [
        "C:\\entrypoint.ps1 -ConfigFile C:\\ecs_windows_forward_daemon\
\cloudwatch.conf"
      ],
      "environment": [
        {
          "name": "AWS_REGION",
          "value": "region"
        }
      ],
      "memory": 512,
      "essential": true,
      "logConfiguration": {
        "logDriver": "awslogs",
        "options": {
          "awslogs-group": "/ecs/fluent-bit-logs",
          "awslogs-region": "region",
          "awslogs-stream-prefix": "flb",
          "awslogs-create-group": "true"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Utilisez la commande suivante pour enregistrer la définition de tâche.

```
aws ecs register-task-definition --cli-input-json file://fluent-bit.json --
region region
```

Vous pouvez répertorier les définitions de tâche de votre compte en exécutant la commande `list-task-definitions`. La sortie affiche les valeurs de famille et de révision que vous pouvez utiliser conjointement avec `run-task` ou `start-task`.

Étape 5 : Exécuter la définition de `tâche ecs-windows-fluent-bit` en tant que service Amazon ECS à l'aide de la stratégie de planification des démons

Après avoir enregistré une définition de tâche pour votre compte, vous pouvez exécuter une tâche dans le cluster. Pour ce didacticiel, vous exécutez une instance de la définition de tâche `ecs-windows-fluent-bit:1` de votre cluster `FluentBit-cluster`. Exécutez la tâche dans un service qui utilise la stratégie de planification du démon, qui garantit qu'une seule instance de Fluent Bit s'exécute toujours sur chacune de vos instances de conteneur.

Pour exécuter une tâche

1. Exécutez la commande suivante pour démarrer la définition de la tâche `ecs-windows-fluent-bit:1` (enregistrée à l'étape précédente) en tant que service.

   > **Note**
   >
   > Cette définition de tâche utilise le pilote de journalisation `awslogs`. Votre instance de conteneur doit disposer des autorisations nécessaires.

   Remplacez les variables suivantes :

   - *`region`*avec la région où fonctionne votre service

   ```
   aws ecs create-service \
       --cluster FluentBit-cluster \
       --service-name FluentBitForwardDaemonService \
       --task-definition ecs-windows-fluent-bit:1 \
       --launch-type EC2 \
       --scheduling-strategy DAEMON \
       --region region
   ```

2. Exécutez la commande suivante pour répertorier vos tâches.

   Remplacez les variables suivantes :

   - *`region`*avec la région dans laquelle s'exécutent vos tâches de service

```
aws ecs list-tasks --cluster FluentBit-cluster --region region
```

Étape 6 : Enregistrer une définition de tâche Windows qui génère les journaux

Enregistrez une définition de tâche qui génère les journaux. Cette définition de tâche déploie une image de conteneur Windows qui écrira un nombre incrémentiel à `stdout` toutes les secondes.

La définition de la tâche utilise le pilote de journalisation Fluentd qui se connecte au port 24224 que le plug-in Fluent Bit écoute. L'agent Amazon ECS met des étiquettes à chaque conteneur ECS qui incluent le nom du cluster, l'ARN de la tâche, le nom de famille de la définition de la tâche, le numéro de révision de la définition de la tâche et le nom du conteneur de la tâche. Ces étiquettes clé-valeur sont transmises à Fluent Bit.

> **Note**
>
> Cette tâche utilise également le mode réseau `default`. Toutefois, vous pouvez aussi utiliser le mode réseau `awsvpc` avec la tâche.

Pour enregistrer une définition de tâche

1. Créez un fichier nommé `windows-app-task.json` avec les contenus suivants.

```
{
  "family": "windows-app-task",
  "containerDefinitions": [
    {
      "name": "sample-container",
      "image": "mcr.microsoft.com/windows/servercore:ltsc2019",
      "cpu": 512,
      "memory": 512,
      "essential": true,
      "entryPoint": [
        "Powershell",
        "-Command"
      ],
```

```
      "command": [
        "$count=1;while(1) { Write-Host $count; sleep 1; $count=$count+1;}"
      ],
      "logConfiguration": {
        "logDriver": "fluentd",
        "options": {
          "fluentd-address": "localhost:24224",
          "tag": "{{ index .ContainerLabels \"com.amazonaws.ecs.task-definition-
family\" }}",
          "fluentd-async": "true",
          "labels": "com.amazonaws.ecs.cluster,com.amazonaws.ecs.container-
name,com.amazonaws.ecs.task-arn,com.amazonaws.ecs.task-definition-
family,com.amazonaws.ecs.task-definition-version"
        }
      }
    }
  ],
  "memory": "512",
  "cpu": "512"
}
```

2. Utilisez la commande suivante pour enregistrer la définition de tâche.

   Remplacez les variables suivantes :

   - *region*avec la région dans laquelle s'exécute votre tâche

   ```
   aws ecs register-task-definition --cli-input-json file://windows-app-task.json --
   region region
   ```

   Vous pouvez répertorier les définitions de tâche de votre compte en exécutant la commande `list-task-definitions`. La sortie affiche les valeurs de famille et de révision que vous pouvez utiliser conjointement avec `run-task` ou `start-task`.

Étape 7 : Exécuter la définition de la tâche **windows-app-task**

Après avoir enregistré la définition de tâche `windows-app-task`, exécutez-la dans votre cluster `FluentBit-cluster`.

Pour exécuter une tâche

1. Exécutez la définition de tâche `windows-app-task:1` que vous avez enregistrée à l'étape précédente.

   Remplacez les variables suivantes :

   - *`region`*avec la région dans laquelle s'exécute votre tâche

   ```
   aws ecs run-task --cluster FluentBit-cluster --task-definition windows-app-task:1
    --count 2 --region region
   ```

2. Exécutez la commande suivante pour répertorier vos tâches.

   ```
   aws ecs list-tasks --cluster FluentBit-cluster
   ```

Étape 8 : Vérifiez les connexions CloudWatch

Afin de vérifier votre configuration Fluent Bit, vérifiez les groupes de journaux suivants dans la CloudWatch console :

- `/ecs/fluent-bit-logs` : il s'agit du groupe de journaux qui correspond au conteneur de démon Fluent Bit qui s'exécute sur l'instance de conteneur.
- `/aws/ecs/FluentBit-cluster.windows-app-task` : il s'agit du groupe de journaux qui correspond à toutes les tâches lancées pour la famille de définition de tâche `windows-app-task` au sein du cluster `FluentBit-cluster`.

  `task-out.`*`FIRST_TASK_ID`*`.sample-container` : ce flux de journaux contient tous les journaux générés par la première instance de la tâche dans le conteneur de tâches sample-container.

  `task-out.`*`SECOND_TASK_ID`*`.sample-container` : ce flux de journaux contient tous les journaux générés par la seconde instance de la tâche dans le conteneur de tâches sample-container.

Le flux de journaux `task-out.`*`TASK_ID`*`.sample-container` contient des champs similaires aux suivants :

```
{
    "source": "stdout",
    "ecs_task_arn": "arn:aws:ecs:region:0123456789012:task/FluentBit-
cluster/13EXAMPLE",
    "container_name": "/ecs-windows-app-task-1-sample-container-cEXAMPLE",
    "ecs_cluster": "FluentBit-cluster",
    "ecs_container_name": "sample-container",
    "ecs_task_definition_version": "1",
    "container_id": "61f5e6EXAMPLE",
    "log": "10",
    "ecs_task_definition_family": "windows-app-task"
}
```

Pour vérifier la configuration Fluent Bit

1. Ouvrez la CloudWatch console à l'adresse https://console.aws.amazon.com/cloudwatch/.
2. Dans le panneau de navigation, choisissez Groupes de journaux. Assurez-vous que vous êtes dans la région où vous avez déployé Fluent Bit sur vos conteneurs.

   Dans la liste des groupes de journaux du Région AWS, vous devriez voir ce qui suit :

   - `/ecs/fluent-bit-logs`
   - `/aws/ecs/FluentBit-cluster.windows-app-task`

   Si vous voyez ces groupes de journaux, la vérification de la configuration Fluent Bit est terminée.

Étape 9 : nettoyer

Lorsque vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Pour nettoyer les ressources du didacticiel

1. Arrêtez la tâche `windows-simple-task` et la tâche `ecs-fluent-bit`. Pour de plus amples informations, veuillez consulter the section called “Arrêt d'une tâche”.
2. Utilisez la commande suivante pour supprimer le groupe de journaux `/ecs/fluent-bit-logs`. Pour plus d'informations sur la suppression de groupes de journaux, voir delete-log-group dans le Guide de référence d'AWS Command Line Interface .

```
aws logs delete-log-group --log-group-name /ecs/fluent-bit-logs
aws logs delete-log-group --log-group-name /aws/ecs/FluentBit-cluster.windows-app-task
```

3. Exécutez la commande suivante pour mettre fin à l'instance.

```
aws ec2 terminate-instances --instance-ids instance-id
```

4. Exécutez les commandes suivantes pour supprimer les rôles IAM.

```
aws iam delete-role --role-name ecsInstanceRole
aws iam delete-role --role-name fluentTaskRole
```

5. Exécutez la commande suivante pour supprimer le cluster Amazon ECS.

```
aws ecs delete-cluster --cluster FluentBit-cluster
```

Utilisation gMSA pour EC2 Linux conteneurs sur Amazon ECS

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux sur EC2 par l'intermédiaire d'un type de compte de service spécial appelé group Managed Service Account (gMSA).

Les applications réseau Linux, comme les applications .NET Core, peuvent faire appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et les services. Vous pouvez utiliser cette fonctionnalité en concevant des applications qui s'intègrent à Active Directory et s'exécutent sur des serveurs joints à un domaine. Toutefois, comme les conteneurs Linux ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Linux pour qu'il soit exécuté avec gMSA.

Un conteneur Linux qui s'exécute avec gMSA repose sur le démon `credentials-fetcher` qui s'exécute sur l'instance Amazon EC2 hôte du conteneur. En d'autres termes, le démon récupère les informations d'identification gMSA auprès du contrôleur de domaine Active Directory, puis les transfère à l'instance de conteneur. Pour plus d'informations sur les comptes de service, veuillez consulter Créer des gMSAs pour des conteneurs Windows sur le site Web de Microsoft Learn.

Considérations

Tenez compte des points suivants avant d'utiliser des gMSA pour les conteneurs Linux :

- Si vos conteneurs fonctionnent sur EC2, vous pouvez utiliser des gMSA pour les conteneurs Windows et Linux. Pour plus d'informations sur l'utilisation de gMSA pour un conteneur Linux sur Fargate, consultez la section Utilisation gMSA for Linux conteneurs sur Fargate.
- Vous aurez peut-être besoin d'un ordinateur Windows joint au domaine pour remplir les conditions requises. Par exemple, vous pourriez avoir besoin d'un ordinateur Windows joint au domaine pour créer le gMSA dans Active Directory avec PowerShell. Les PowerShell outils RSAT Active Director ne sont disponibles que pourWindows. Pour plus d'informations, veuillez consulter Installation des outils d'administration Active Directory (langue française non garantie).
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

  Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine

Emplacement de stockage	Linux	Windows
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

Conditions préalables

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Linux avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un Directory Service Active Directory. Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour de plus amples informations, veuillez consulter Utilisation gMSA pour EC2 Linux conteneurs sur Amazon ECS.
- Vous avez installé et vous exécutez le démon `credentials-fetcher` sur une instance de conteneur Linux Amazon ECS. Vous avez également ajouté un jeu initial d'informations d'identification au démon `credentials-fetcher` pour vous authentifier auprès d'Active Directory.

  > **Note**
  >
  > Le démon `credentials-fetcher` est uniquement disponible pour Amazon Linux 2023 et Fedora 37 et versions ultérieures. Le démon n'est pas disponible pour Amazon Linux 2. Pour plus d'informations, consultez aws/credentials-fetcher on. GitHub

- Vous configurez les informations d'identification permettant au démon `credentials-fetcher` de s'authentifier auprès d'Active Directory. Les informations d'identification doivent être membres

du groupe de sécurité Active Directory qui a accès au compte gMSA. Il existe plusieurs options dans Décidez si vous souhaitez joindre les instances au domaine ou utiliser le mode sans domaine gMSA..

- Vous avez ajouté les autorisations IAM requises. Les autorisations requises dépendent des méthodes que vous choisissez pour les informations d'identification initiales et pour le stockage de la spécification des informations d'identification :
  - Si vous utilisez domainless gMSA pour les informations d'identification initiales, les autorisations IAM pour AWS Secrets Manager sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans SSM Parameter Store, les autorisations IAM d'Amazon EC2 Systems Manager Parameter Store sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans Amazon S3, les autorisations IAM d'Amazon Simple Storage Service sont requises sur le rôle d'exécution de la tâche.

Configuration gMSA-capable Linux Conteneurs sur Amazon ECS

Préparation de l'infrastructure

Les étapes suivantes sont des considérations et une configuration qui ne sont effectuées qu'une seule fois. Après avoir terminé ces étapes, vous pouvez automatiser la création d'instances de conteneur afin de réutiliser cette configuration.

Décidez de la manière dont les informations d'identification initiales sont fournies et configurez les données utilisateur EC2 dans un modèle de lancement EC2 réutilisable pour installer le démon `credentials-fetcher`.

1. Décidez si vous souhaitez joindre les instances au domaine ou utiliser le mode sans domaine gMSA.
   - Jonction des instances EC2 au domaine Active Directory
     - Jointure des instances par les données utilisateur

       Ajoutez les étapes permettant de joindre le domaine Active Directory à vos données utilisateur EC2 dans un modèle de lancement EC2. Plusieurs groupes Amazon EC2 Auto Scaling peuvent utiliser le même modèle de lancement.

Vous pouvez suivre ces étapes pour Rejoindre un Active Directory ou un domaine FreeIPA dans les Fedora Docs (langue française non garantie).

- Créer un utilisateur Active Directory pour domainless gMSA

  Le démon `credentials-fetcher` possède une fonctionnalité appelée gMSA sans domaine. Cette fonctionnalité nécessite un domaine, mais il n'est pas nécessaire que l'instance EC2 soit jointe au domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance. Vous devez plutôt fournir le nom d'un secret d' AWS Secrets Manager dans le fichier CredSpec. Le secret doit contenir un nom d'utilisateur, un mot de passe et le domaine auquel se connecter.

  Cette fonctionnalité est prise en charge et peut être utilisée avec les conteneurs Linux et Windows.

  Cette fonctionnalité est similaire à la fonctionnalité gMSA support for non-domain-joined container hosts. Pour plus d'informations sur la fonctionnalité Windows, veuillez consulter Architecture et améliorations des gMSA sur le site Web de Microsoft Learn.

  a. Créez un utilisateur dans votre domaine Active Directory. L'utilisateur d'Active Directory doit être autorisé à accéder aux comptes de service gMSA que vous utilisez dans les tâches.

  b. Créez un secret dans AWS Secrets Manager, après avoir créé l'utilisateur dans Active Directory. Pour plus d'informations, voir Création d'un AWS Secrets Manager secret.

  c. Saisissez respectivement le nom d'utilisateur, le mot de passe et le domaine de l'utilisateur dans les paires clé-valeur JSON appelées `username`, `password` et `domainName`.

  ```
  {"username":"username","password":"passw0rd", "domainName":"example.com"}
  ```

  d. Ajoutez la configuration au fichier CredSpec pour le compte de service. La `HostAccountConfig` supplémentaire contient l'Amazon Resource Name (ARN) du secret dans Secrets Manager.

Sous Windows, le `PluginGUID` doit correspondre au GUID indiqué dans l'exemple de code suivant. Sous Linux, le `PluginGUID` est ignoré. Remplacez `MySecret` par l'exemple avec l'Amazon Resource Name (ARN) de votre secret.

```
    "ActiveDirectoryConfig": {
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
```

e. La fonctionnalité gMSA sans domaine nécessite des autorisations supplémentaires dans le rôle d'exécution des tâches. Suivez l'étape (Facultatif) sans domaine gMSA secret.

2. Configuration des instances et installation du démon **`credentials-fetcher`**

Vous pouvez installer le démon `credentials-fetcher` avec un script de données utilisateur dans votre modèle de lancement EC2. Les exemples suivants illustrent deux types de données utilisateur, `cloud-config` YAML ou le script bash. Ces exemples concernent Amazon Linux 2023 (AL2023). Remplacez `MyCluster` par le nom du cluster Amazon ECS que vous souhaitez que ces instances rejoignent.

- **`configuration du cloud`** YAML

```
Content-Type: text/cloud-config
package_reboot_if_required: true
packages:
  # prerequisites
  - dotnet
  - realmd
  - oddjob
  - oddjob-mkhomedir
  - sssd
  - adcli
  - krb5-workstation
  - samba-common-tools
  # https://github.com/aws/credentials-fetcher gMSA credentials management for
 containers
```

```
  - credentials-fetcher
write_files:
# configure the ECS Agent to join your cluster.
# replace MyCluster with the name of your cluster.
- path: /etc/ecs/ecs.config
  owner: root:root
  permissions: '0644'
  content: |
    ECS_CLUSTER=MyCluster
    ECS_GMSA_SUPPORTED=true
runcmd:
# start the credentials-fetcher daemon and if it succeeded, make it start after
 every reboot
- "systemctl start credentials-fetcher"
- "systemctl is-active credentials-fetcher && systemctl enable credentials-
fetcher"
```

- bash script

  Si vous êtes plus à l'aise avec les scripts bash et que vous avez plusieurs variables à écrire dans `/etc/ecs/ecs.config`, utilisez le format `heredoc` suivant. Ce format écrit tout entre les lignes commençant par cat et `EOF` dans le fichier de configuration.

```
#!/usr/bin/env bash
set -euxo pipefail

# prerequisites
timeout 30 dnf install -y dotnet realmd oddjob oddjob-mkhomedir sssd adcli
 krb5-workstation samba-common-tools
# install https://github.com/aws/credentials-fetcher gMSA credentials
 management for containers
timeout 30 dnf install -y credentials-fetcher

# start credentials-fetcher
systemctl start credentials-fetcher
systemctl is-active credentials-fetcher && systemctl enable credentials-fetcher

cat <<'EOF' >> /etc/ecs/ecs.config
ECS_CLUSTER=MyCluster
ECS_GMSA_SUPPORTED=true
EOF
```

Il existe des variables de configuration facultatives pour le démon `credentials-fetcher` que vous pouvez définir dans `/etc/ecs/ecs.config`. Nous vous recommandons de définir les variables dans les données utilisateur du bloc YAML ou `heredoc` de manière similaire aux exemples précédents. Cela permet d'éviter les problèmes de configuration partielle qui peuvent survenir lors de la modification d'un fichier à plusieurs reprises. Pour plus d'informations sur la configuration de l'agent ECS, consultez Amazon ECS Container Agent on GitHub.

- Vous pouvez éventuellement utiliser la variable `CREDENTIALS_FETCHER_HOST` si vous modifiez la configuration du démon `credentials-fetcher` pour déplacer le socket vers un autre emplacement.

Configuration des autorisations et des secrets

Effectuez les étapes suivantes une fois pour chaque application et chaque définition de tâche. Nous vous recommandons d'utiliser le principe de moindre privilège et d'affiner les autorisations utilisées dans la stratégie. Ainsi, chaque tâche ne peut lire que les secrets dont elle a besoin.

1. (Facultatif) sans domaine gMSA secret

   Si vous utilisez la méthode sans domaine dans laquelle l'instance n'est pas jointe au domaine, procédez comme suit.

   Vous devez ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche. Cela permet au démon `credentials-fetcher` d'accéder au secret Secrets Manager. Remplacez l'exemple `MySecret` par l'Amazon Resource Name (ARN) de votre secret dans la liste `Resource`.

   JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:123456789012:secret:my-secret-AbCdEf"
```

```
        }
    ]
}
```

> **Note**
>
> Si vous utilisez votre propre clé KMS pour chiffrer votre secret, vous devez ajouter les autorisations nécessaires à ce rôle et ajouter ce rôle à la politique des AWS KMS clés.

2. Décidez si vous utilisez le magasin de paramètres SSM ou S3 pour stocker le CredSpec

   Amazon ECS permet de référencer le chemin du fichier dans le champ `credentialSpecs` d'une définition de tâche.

   Si vous joignez les instances à un seul domaine, utilisez le préfixe `credentialspec:` au début de l'ARN dans la chaîne. Si vous utilisez le gMSA sans domaine, utilisez `credentialspecdomainless:`.

   Pour en savoir plus sur CredSpec, consultez Fichier de spécification des informations d'identification.

   - Compartiment Amazon S3

     Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3. Référencez ensuite l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/
${ObjectName}"
            ],
            ...
        }
    ],
```

```
    ...
}
```

Pour permettre à vos tâches d'accéder au compartiment S3, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::amzn-s3-demo-bucket",
                "arn:aws:s3:::amzn-s3-demo-bucket/{object}"
            ]
        }
    ]
}
```

- Paramètre SSM Parameter Store

  Ajoutez la spécification d'informations d'identification à un paramètre de SSM Parameter Store. Référencez ensuite l'Amazon Resource Name (ARN) du paramètre de SSM Parameter Store dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
```

```
                "credentialspecdomainless:arn:aws:ssm:aws-
region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Pour permettre à vos tâches d'accéder au paramètre de SSM Parameter Store, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
            "Resource": "arn:aws:ssm:us-east-1:123456789012:parameter/my-
parameter"
        }
    ]
}
```

Fichier de spécification des informations d'identification

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Linux. Vous générez le CredSpec et le référencez dans le champ `credentialSpecs` de votre définition de tâche. Le fichier CredSpec ne contient aucun secret.

Voici un exemple de fichier CredSpec.

```
{
    "CmsPlugins": [
```

```
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Création d'un CredSpec

Vous créez un CredSpec en utilisant le module PowerShell CredSpec sur un ordinateur Windows joint au domaine. Suivez les étapes décrites dans Créer une spécification d'informations d'identification sur le site Web Microsoft Learn.

Utilisation gMSA for Linux conteneurs sur Fargate

Amazon ECS prend en charge l’authentification Active Directory pour les conteneurs Linux sur Fargate par l’intermédiaire d’un type de compte de service spécial appelé compte de service géré de groupe (gMSA).

Les applications réseau Linux, comme les applications .NET Core, peuvent faire appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et

les services. Vous pouvez utiliser cette fonctionnalité en concevant des applications qui s'intègrent à Active Directory et s'exécutent sur des serveurs joints à un domaine. Toutefois, comme les conteneurs Linux ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Linux pour qu'il soit exécuté avec gMSA.

Considérations

Tenez compte des points suivants avant d'utiliser gMSA pour les conteneurs Linux sur Fargate :

- Vous devez exécuter la version de plateforme 1.4 ou ultérieure.
- Vous aurez peut-être besoin d'un ordinateur Windows joint au domaine pour remplir les conditions requises. Par exemple, vous pourriez avoir besoin d'un ordinateur Windows joint au domaine pour créer le gMSA dans Active Directory avec PowerShell. Les PowerShell outils RSAT Active Director ne sont disponibles que pourWindows. Pour plus d'informations, veuillez consulter Installation des outils d'administration Active Directory (langue française non garantie).
- Vous devez utiliser l'option gMSA sans domaine

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans un compartiment Amazon S3.
- Une tâche ne peut prendre en charge qu'un seul Active Directory.

Conditions préalables

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Linux avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un Directory Service Active Directory. Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.

- Vous disposez d’un compte gMSA existant dans Active Directory et d’un utilisateur autorisé à accéder au compte de service gMSA. Pour de plus amples informations, veuillez consulter Créer un utilisateur Active Directory pour domainless gMSA.
- Vous disposez d’un compartiment Amazon S3. Pour plus d’informations, consultez la section Création d’un compartiment dans le Guide de l’utilisateur Amazon S3.

Configuration gMSA-capable Linux Conteneurs sur Amazon ECS

Préparation de l'infrastructure

Les étapes suivantes sont des considérations et une configuration qui ne sont effectuées qu'une seule fois.

- Créer un utilisateur Active Directory pour domainless gMSA

  Lorsque vous utilisez l’option gMSA sans domaine, le conteneur n’est pas joint au domaine. Les autres applications qui s’exécutent sur le conteneur ne peuvent pas utiliser les informations d’identification pour accéder au domaine. Les tâches qui utilisent un domaine différent peuvent être exécutées sur le même conteneur. Vous indiquez le nom d'un secret AWS Secrets Manager dans le CredSpec fichier. Le secret doit contenir un nom d'utilisateur, un mot de passe et le domaine auquel se connecter.

  Cette fonctionnalité est similaire à la fonctionnalité gMSA support for non-domain-joined container hosts. Pour plus d'informations sur la fonctionnalité Windows, veuillez consulter Architecture et améliorations des gMSA sur le site Web de Microsoft Learn.

  a. Configurez un utilisateur dans votre domaine Active Directory. L’utilisateur d’Active Directory doit être autorisé à accéder au compte de service gMSA que vous utilisez dans les tâches.
  b. Vous disposez d'un VPC et de sous-réseaux capables de résoudre le nom de domaine Active Directory. Configurez le VPC avec les options DHCP en utilisant le nom de domaine qui pointe vers le nom du service Active Directory. Pour plus d’informations sur les options DHCP, consultez la section Utilisation des jeux d’options DHCP dans le Guide de l’utilisateur Amazon Virtual Private Cloud.
  c. Créez un secret dans AWS Secrets Manager.
  d. Créez le fichier de spécification des informations d’identification.

Configuration des autorisations et des secrets

Effectuez les étapes suivantes une fois pour chaque application et chaque définition de tâche. Nous vous recommandons d'utiliser le principe de moindre privilège et d'affiner les autorisations utilisées dans la stratégie. Ainsi, chaque tâche ne peut lire que les secrets dont elle a besoin.

1. Créez un utilisateur dans votre domaine Active Directory. L'utilisateur d'Active Directory doit être autorisé à accéder aux comptes de service gMSA que vous utilisez dans les tâches.
2. Après avoir créé l'utilisateur Active Directory, créez un secret dans AWS Secrets Manager. Pour plus d'informations, veuillez consulter Créer un secret AWS Secrets Manager.
3. Saisissez respectivement le nom d'utilisateur, le mot de passe et le domaine de l'utilisateur dans les paires clé-valeur JSON appelées `username`, `password` et `domainName`.

   ```
   {"username":"username","password":"passw0rd", "domainName":"example.com"}
   ```

4. Vous devez ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche. Cela permet au démon `credentials-fetcher` d'accéder au secret Secrets Manager. Remplacez l'exemple `MySecret` par l'Amazon Resource Name (ARN) de votre secret dans la liste `Resource`.

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Action": [
                   "secretsmanager:GetSecretValue"
               ],
               "Resource": [
               "arn:aws:secretsmanager:us-east-1:111122223333:secret:MySecret"
               ]
           }
       ]
   }
   ```

> **Note**
>
> Si vous utilisez votre propre clé KMS pour chiffrer votre secret, vous devez ajouter les autorisations nécessaires à ce rôle et ajouter ce rôle à la politique des AWS KMS clés.

5. Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3. Référencez ensuite l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Pour permettre à vos tâches d'accéder au compartiment S3, ajoutez les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
            "s3:GetObjectVersion",
            "s3:ListBucket"
```

```
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
        }
    ]
}
```

Fichier de spécification des informations d'identification

Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Linux. Vous générez le CredSpec et le référencez dans le champ `credentialSpecs` de votre définition de tâche. Le fichier CredSpec ne contient aucun secret.

Voici un exemple de fichier CredSpec.

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "example.com",
        "DnsName": "example.com",
        "NetBiosName": "example"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "example.com"
            }
        ],
        "HostAccountConfig": {
            "PortableCcgVersion": "1",
            "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
            "PluginInput": {
```

```
                "CredentialArn": "arn:aws:secretsmanager:aws-
region:111122223333:secret:MySecret"
            }
        }
    }
}
```

Création d'un CredSpec et en le téléchargeant sur un Amazon S3

Vous créez un CredSpec en utilisant le module PowerShell CredSpec sur un ordinateur Windows joint au domaine. Suivez les étapes décrites dans Créer une spécification d'informations d'identification sur le site Web Microsoft Learn.

Après avoir créé le fichier de spécification des informations d'identification, chargez-le dans un compartiment Amazon S3. Copiez le fichier CredSpec sur l'ordinateur ou l'environnement dans lequel vous exécutez les commandes AWS CLI .

Exécutez la AWS CLI commande suivante pour le CredSpec télécharger sur Amazon S3. Remplacez *amzn-s3-demo-bucket* par le nom de votre compartiment Simple Storage Service (Amazon S3). Vous pouvez stocker le fichier sous forme d'objet dans n'importe quel compartiment et à n'importe quel emplacement, mais vous devez autoriser l'accès à ce compartiment et à cet emplacement dans la stratégie que vous associez au rôle d'exécution des tâches.

Pour PowerShell, utilisez la commande suivante :

```
$ Write-S3Object -BucketName "amzn-s3-demo-bucket" -Key "ecs-domainless-gmsa-credspec"
 -File "gmsa-cred-spec.json"
```

La AWS CLI commande suivante utilise des barres obliques inverses qui sont utilisées par sh les interpréteurs de commandes compatibles.

```
$ aws s3 cp gmsa-cred-spec.json \
s3://amzn-s3-demo-bucket/ecs-domainless-gmsa-credspec
```

Utilisation de conteneurs Windows Amazon ECS avec un environnement sans domaine gMSA en utilisant le AWS CLI

Le didacticiel suivant explique comment créer une tâche Amazon ECS qui exécute un conteneur Windows qui possède des informations d'identification pour accéder à Active Directory avec l' AWS

CLI. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Rubriques

- Conditions préalables
- Étape 1 : Création et configuration du gMSA compte sur les services de domaine Active Directory (AD DS)
- Étape 2 : charger les informations d'identification dans Secrets Manager
- Étape 3 : Modifiez votre CredSpec JSON à inclure sans domaine gMSA Informations
- Étape 4 : Téléverser CredSpec vers Amazon S3
- Étape 5 (facultative) : créer un cluster Amazon ECS
- Étape 6 : créer un rôle IAM pour les instances de conteneur
- Étape 7 : créer un rôle d'exécution de tâche personnalisé
- Étape 8 : créer un rôle de tâche pour Amazon ECS Exec
- Étape 9 : enregistrer une définition de tâche qui utilise le mode sans domaine gMSA
- Étape 10 : enregistrer une instance de conteneur Windows dans le cluster
- Étape 11 : vérifier l'instance de conteneur
- Étape 12 : exécuter une tâche Windows
- Étape 13 : Vérifiez que le conteneur contient gMSA informations d’identification
- Étape 14 : nettoyer
- Débogage d'Amazon ECS sans domaine gMSA pour conteneurs Windows

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l’exemple de politique IAM Amazon ECS_FullAccess.
- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau de votre AWS CLI AWS Command Line Interface.

> **Note**
>
> Vous pouvez utiliser des points de terminaison de service à double pile pour interagir avec Amazon ECS à partir des AWS CLI SDK et de l'API Amazon ECS sur IPv4 et IPv6. Pour de plus amples informations, veuillez consulter Utilisation des points de terminaison à double pile Amazon ECS.

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un Directory Service Active Directory. Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un VPC et de sous-réseaux capables de résoudre le nom de domaine Active Directory.
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

  Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

- (Facultatif) AWS CloudShell est un outil qui fournit aux clients une ligne de commande sans avoir à créer leur propre instance EC2. Pour plus d'informations, voir Qu'est-ce que c'est AWS CloudShell ? dans le guide de AWS CloudShell l'utilisateur.

Étape 1 : Création et configuration du gMSA compte sur les services de domaine Active Directory (AD DS)

Créez et configurez un compte gMSA sur le domaine Active Directory.

> **Note**
>
> Cette étape crée deux comptes distincts : un compte de service géré de groupe (gMSA) qui fournit l'identité de vos conteneurs, et un compte utilisateur normal utilisé pour l'authentification de domaine. Ces comptes ont des objectifs différents et doivent porter des noms différents.

1. Génération d'une clé racine du service de diffusion de clés

   > **Note**
   >
   > Si vous utilisez Directory Service, vous pouvez ignorer cette étape.

La clé racine KDS et les autorisations gMSA sont configurées avec votre AD Microsoft géré par AWS .

Si vous n'avez pas encore créé de compte de service gMSA dans votre domaine, vous devez d'abord générer une clé racine du service de diffusion de clés (KDS). Le KDS est responsable de la création, de la rotation et de la diffusion du mot de passe gMSA aux hôtes autorisés. Lorsque `ccg.exe` a besoin de récupérer les informations d'identification gMSA, il contacte KDS pour récupérer le mot de passe actuel.

Pour vérifier si la clé racine KDS a déjà été créée, exécutez l' PowerShellapplet de commande suivante avec les privilèges d'administrateur de domaine sur un contrôleur de domaine à l'aide du module. `ActiveDirectory` PowerShell Pour plus d'informations sur le module, voir ActiveDirectory Module sur le site Web de Microsoft Learn.

```
PS C:\> Get-KdsRootKey
```

Si la commande renvoie un ID de clé, vous pouvez ignorer le reste de cette étape. Dans le cas contraire, créez la clé racine KDS en exécutant la commande suivante :

```
PS C:\> Add-KdsRootKey -EffectiveImmediately
```

Bien que l'argument `EffectiveImmediately` de la commande implique que la clé est effective immédiatement, vous devez attendre 10 heures avant que la clé racine KDS soit répliquée et puisse être utilisée sur tous les contrôleurs de domaine.

2. Créez le gMSA compte

Pour créer le gMSA compte et autoriser le `ccg.exe` à récupérer le gMSA mot de passe, exécutez les PowerShell commandes suivantes à partir d'un serveur ou d'un client Windows ayant accès au domaine. Remplacez `ExampleAccount` par le nom que vous souhaitez pour votre compte gMSA et remplacez `example-domain` par votre nom de domaine Active Directory (par exemple, si votre domaine est `contoso.com`, utilisez `contoso`).

a.
```
PS C:\> Install-WindowsFeature RSAT-AD-PowerShell
```

b.
```
PS C:\> New-ADGroup -Name "ExampleAccount Authorized Hosts" -SamAccountName
 "ExampleAccountHosts" -GroupScope DomainLocal
```

c.
```
PS C:\> New-ADServiceAccount -Name "ExampleAccount" -DnsHostName "example-
domain" -ServicePrincipalNames "host/ExampleAccount", "host/example-domain" -
PrincipalsAllowedToRetrieveManagedPassword "ExampleAccountHosts"
```

d. Créez un utilisateur avec un mot de passe permanent qui n'expire pas. Ces informations d'identification sont stockées AWS Secrets Manager et utilisées par chaque tâche pour rejoindre le domaine. Il s'agit d'un compte utilisateur distinct du compte GMSA créé précédemment. Remplacez `ExampleServiceUser` par le nom que vous souhaitez pour ce compte utilisateur de service.

```
PS C:\> New-ADUser -Name "ExampleServiceUser" -AccountPassword (ConvertTo-
SecureString -AsPlainText "Test123" -Force) -Enabled 1 -PasswordNeverExpires 1
```

e.
```
PS C:\> Add-ADGroupMember -Identity "ExampleAccountHosts" -Members
 "ExampleServiceUser"
```

f. Installez le PowerShell module de création d'CredSpecobjets dans Active Directory et générez le CredSpec JSON.

```
PS C:\> Install-PackageProvider -Name NuGet -Force
```

```
PS C:\> Install-Module CredentialSpec
```

g.
```
PS C:\> New-CredentialSpec -AccountName ExampleAccount
```

3. Copiez le résultat JSON de la commande précédente dans un fichier appelé `gmsa-cred-spec.json`. Il s'agit du fichier CredSpec. Il est utilisé à l'étape 3, Étape 3 : Modifiez votre CredSpec JSON à inclure sans domaine gMSA Informations.

Étape 2 : charger les informations d'identification dans Secrets Manager

Copiez les informations d'identification Active Directory dans un système de stockage d'informations d'identification sécurisé, afin que chaque tâche les récupère. Il s'agit de la méthode gMSA sans domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

- Exécutez la AWS CLI commande suivante et remplacez le nom d'utilisateur, le mot de passe et le nom de domaine en fonction de votre environnement. Utilisez le nom du compte utilisateur du service (et non le nom du compte gMSA) comme nom d'utilisateur. Conservez l'ARN du secret pour l'utiliser à l'étape suivante, Étape 3 : Modifiez votre CredSpec JSON à inclure sans domaine gMSA Informations.

  La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws secretsmanager create-secret \
--name gmsa-plugin-input \
--description "Amazon ECS - gMSA Portable Identity." \
--secret-string "{\"username\":\"ExampleServiceUser\",\"password\":\"Test123\",
\"domainName\":\"contoso.com\"}"
```

Étape 3 : Modifiez votre CredSpec JSON à inclure sans domaine gMSA Informations

Avant de charger le CredSpec vers l'une des options de stockage, ajoutez des informations au CredSpec à l'aide de l'ARN du secret dans Secrets Manager provenant de l'étape précédente. Pour plus d'informations, veuillez consulter Configuration de spécifications d'informations d'identification supplémentaires pour un cas d'usage avec un hôte de conteneur non joint à un domaine sur le site Web de Microsoft Learn.

1. Ajoutez les informations suivantes au fichier CredSpec contenu dans le `ActiveDirectoryConfig`. Remplacez l'ARN par le secret de l'étape précédente dans Secrets Manager.

   Veuillez noter que la valeur `PluginGUID` doit correspondre au GUID de l'extrait de code suivant et qu'elle est obligatoire.

```
"HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
```

```
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
```

Vous pouvez également utiliser un secret dans SSM Parameter Store en utilisant l'ARN au format suivant : `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

2. Une fois le fichier CredSpec modifié, il doit ressembler à l'exemple suivant :

```
{
  "CmsPlugins": [
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "ExampleAccount",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "example-domain",
    "DnsName": "example-domain",
    "NetBiosName": "example-domain"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      },
      {
        "Name": "ExampleAccount",
        "Scope": "example-domain"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Étape 4 : Téléverser CredSpec vers Amazon S3

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Copiez le fichier CredSpec sur l'ordinateur ou l'environnement dans lequel vous exécutez les commandes AWS CLI .
2. Exécutez la AWS CLI commande suivante pour le CredSpec télécharger sur Amazon S3. Remplacez `MyBucket` par le nom de votre compartiment Simple Storage Service (Amazon S3). Vous pouvez stocker le fichier sous forme d'objet dans n'importe quel compartiment et à n'importe quel emplacement, mais vous devez autoriser l'accès à ce compartiment et à cet emplacement dans la stratégie que vous associez au rôle d'exécution des tâches.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws s3 cp gmsa-cred-spec.json \
   s3://MyBucket/ecs-domainless-gmsa-credspec
   ```

Étape 5 (facultative) : créer un cluster Amazon ECS

Par défaut, votre compte possède un cluster Amazon ECS nommé `default`. Ce cluster est utilisé par défaut dans AWS CLI les SDK et CloudFormation. Vous pouvez utiliser des clusters supplémentaires pour regrouper et organiser les tâches et l'infrastructure, et attribuer des valeurs par défaut pour certaines configurations.

Vous pouvez créer un cluster à partir des SDK Console de gestion AWS AWS CLI, ou CloudFormation. Les paramètres et la configuration du cluster n'ont aucune incidence sur gMSA.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

```
$ aws ecs create-cluster --cluster-name windows-domainless-gmsa-cluster
```

> **Important**
>
> Si vous choisissez de créer votre propre cluster, vous devez spécifier `--cluster clusterName` pour chaque commande que vous prévoyez d'utiliser avec ce cluster.

Étape 6 : créer un rôle IAM pour les instances de conteneur

Une instance de conteneur est un ordinateur hôte permettant d'exécuter des conteneurs dans des tâches ECS, par exemple des instances Amazon EC2. Chaque instance de conteneur s'enregistre dans un cluster Amazon ECS. Avant de pouvoir lancer des instances Amazon EC2 et de les enregistrer dans un cluster, vous devez créer le rôle IAM que vos instances de conteneur utiliseront.

Pour créer le rôle d'instance de conteneur, veuillez consulter Rôle IAM d'instance de conteneur Amazon ECS. Le `ecsInstanceRole` par défaut dispose des autorisations suffisantes pour terminer ce didacticiel.

Étape 7 : créer un rôle d'exécution de tâche personnalisé

Amazon ECS peut utiliser un rôle IAM différent pour les autorisations nécessaires au démarrage de chaque tâche, au lieu du rôle d'instance de conteneur. Il s'agit du rôle d'exécution de tâche. Nous recommandons de créer un rôle d'exécution de tâche avec uniquement les autorisations requises pour qu'ECS exécute la tâche, également appelées autorisations de moindre privilège. Pour plus d'informations sur le principe du moindre privilège, voir SEC03-BP02Accorder l'accès au moindre privilège dans le AWS Well-ArchitectedFramework.

1. Pour créer un rôle d'exécution de tâche, veuillez consulter Création du rôle d'exécution de tâche. Les autorisations par défaut permettent à l'instance de conteneur d'extraire des images de conteneur d'Amazon Elastic Container Registry `stdout` et `stderr` de vos applications pour les connecter à Amazon CloudWatch Logs.

   Comme le rôle nécessite des autorisations personnalisées pour ce didacticiel, vous pouvez lui donner un nom différent de `ecsTaskExecutionRole`. Ce didacticiel utilise `ecsTaskExecutionRole` dans les étapes suivantes.
2. Ajoutez les autorisations suivantes en créant une stratégie personnalisée, soit une stratégie en ligne qui n'existe que pour ce rôle, soit une stratégie que vous pouvez réutiliser. Remplacez l'ARN de la `Resource` dans la première instruction par le compartiment et l'emplacement Amazon S3, et la seconde `Resource` par l'ARN du secret dans Secrets Manager.

Si vous chiffrez le secret dans Secrets Manager avec une clé personnalisée, vous devez également autoriser `kms:Decrypt` pour la clé.

Si vous utilisez SSM Parameter Store au lieu de Secrets Manager, vous devez autoriser `ssm:GetParameter` pour le paramètre, au lieu de `secretsmanager:GetSecretValue`.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "s3:GetObject"
            ],
            "Resource": "arn:aws:s3:::MyBucket/ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json"
        },
        {
            "Effect": "Allow",
            "Action": [
                "secretsmanager:GetSecretValue"
            ],
            "Resource": "arn:aws:secretsmanager:us-
east-1:111122223333:secret:gmsa-plugin-AbCdEf"
        }
    ]
}
```

Étape 8 : créer un rôle de tâche pour Amazon ECS Exec

Ce didacticiel utilise Amazon ECS Exec pour vérifier le fonctionnement en exécutant une commande dans une tâche en cours d'exécution. Pour utiliser ECS Exec, le service ou la tâche doit activer ECS Exec, et le rôle de tâche (mais pas le rôle d'exécution de tâche) doit disposer d'autorisations `ssmmessages`. Pour la politique IAM requise, veuillez consulter Autorisations ECS Exec.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

Pour créer un rôle de tâche à l'aide de AWS CLI, procédez comme suit.

1. Créez un fichier nommé `ecs-tasks-trust-policy.json` avec le contenu suivant :

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
               "Principal": {
                   "Service": "ecs-tasks.amazonaws.com"
               },
               "Action": "sts:AssumeRole"
           }
       ]
   }
   ```

2. Créez un rôle IAM. Vous pouvez remplacer le nom `ecs-exec-demo-task-role` mais le conserver pour les étapes suivantes.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws iam create-role --role-name ecs-exec-demo-task-role \
   --assume-role-policy-document file://ecs-tasks-trust-policy.json
   ```

   Vous pouvez supprimer le fichier `ecs-tasks-trust-policy.json`.

3. Créez un fichier nommé `ecs-exec-demo-task-role-policy.json` avec le contenu suivant :

   JSON

   ```
   {
       "Version":"2012-10-17",
       "Statement": [
           {
               "Effect": "Allow",
   ```

```
            "Action": [
                "ssmmessages:CreateControlChannel",
                "ssmmessages:CreateDataChannel",
                "ssmmessages:OpenControlChannel",
                "ssmmessages:OpenDataChannel"
            ],
            "Resource": "*"
        }
    ]
}
```

4. Créez une politique IAM et attachez-la au rôle de l'étape précédente.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws iam put-role-policy \
       --role-name ecs-exec-demo-task-role \
       --policy-name ecs-exec-demo-task-role-policy \
       --policy-document file://ecs-exec-demo-task-role-policy.json
   ```

   Vous pouvez supprimer le fichier `ecs-exec-demo-task-role-policy.json`.

Étape 9 : enregistrer une définition de tâche qui utilise le mode sans domaine gMSA

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Créez un fichier nommé `windows-gmsa-domainless-task-def.json` avec le contenu suivant :

   ```
   {
     "family": "windows-gmsa-domainless-task",
     "containerDefinitions": [
       {
         "name": "windows_sample_app",
         "image": "mcr.microsoft.com/windows/servercore/iis",
         "cpu": 1024,
   ```

```
      "memory": 1024,
      "essential": true,
      "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::ecs-domainless-gmsa-
credspec/gmsa-cred-spec.json"
      ],
      "entryPoint": [
        "powershell",
        "-Command"
      ],
      "command": [
        "New-Item -Path C:\\inetpub\\wwwroot\\index.html -ItemType file -Value
 '<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>' -Force ; C:\
\ServiceMonitor.exe w3svc"
      ],
      "portMappings": [
        {
          "protocol": "tcp",
          "containerPort": 80,
          "hostPort": 8080
        }
      ]
    }
  ],
  "taskRoleArn": "arn:aws:iam::111122223333:role/ecs-exec-demo-task-role",
  "executionRoleArn": "arn:aws:iam::111122223333:role/ecsTaskExecutionRole"
}
```

2. Enregistrez la définition de tâche en exécutant la commande suivante :

   La commande suivante utilise des barres obliques inverses qui sont utilisées par sh et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws ecs register-task-definition \
--cli-input-json file://windows-gmsa-domainless-task-def.json
```

Étape 10 : enregistrer une instance de conteneur Windows dans le cluster

Lancez une instance Windows Amazon EC2 et exécutez l'agent de conteneur ECS pour l'enregistrer en tant qu'instance de conteneur dans le cluster. ECS exécute des tâches sur les instances de conteneur enregistrées dans le cluster dans lequel les tâches sont démarrées.

1. Pour lancer une instance Windows Amazon EC2 configurée pour Amazon ECS dans le Console de gestion AWS, consultez. Lancement d'une instance de conteneur Amazon ECS Windows Arrêtez-vous à l'étape des données utilisateur.
2. Pour gMSA, les données utilisateur doivent définir la variable d'environnement `ECS_GMSA_SUPPORTED` avant de démarrer l'agent de conteneur ECS.

   Pour ECS Exec, l'agent doit commencer par l'argument `-EnableTaskIAMRole`.

   Pour sécuriser le rôle IAM de l'instance en empêchant les tâches d'atteindre le service Web EC2 IMDS pour récupérer les informations d'identification du rôle, ajoutez l'argument `-AwsvpcBlockIMDS`. Cela s'applique uniquement aux tâches qui utilisent le mode réseau `awsvpc`.

   ```
   <powershell>
   [Environment]::SetEnvironmentVariable("ECS_GMSA_SUPPORTED", $TRUE, "Machine")
   Import-Module ECSTools
   Initialize-ECSAgent -Cluster windows-domainless-gmsa-cluster -EnableTaskIAMRole -
   AwsvpcBlockIMDS
   </powershell>
   ```
3. Consultez un résumé de la configuration de votre instance dans le panneau Summary (Récapitulatif) et, lorsque vous êtes prêt, choisissez Launch instance (Lancer l'instance).

Étape 11 : vérifier l'instance de conteneur

Vous pouvez vérifier qu'il existe une instance de conteneur dans le cluster à l'aide de l' Console de gestion AWS. Cependant, gMSA nécessite des fonctionnalités supplémentaires indiquées sous forme d'attributs. Ces attributs ne sont pas visibles dans le Console de gestion AWS. Ce didacticiel utilise donc le AWS CLI.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Répertoriez les instances de conteneur dans le cluster. Les instances de conteneur ont un ID différent de celui de l'instance EC2.

   ```
   $ aws ecs list-container-instances
   ```

   Sortie :

   ```
   {
       "containerInstanceArns": [
           "arn:aws:ecs:aws-region:111122223333:container-
   instance/default/MyContainerInstanceID"
       ]
   }
   ```

   Par exemple, `526bd5d0ced448a788768334e79010fd` est un ID d'instance de conteneur valide.

2. Utilisez l'ID d'instance de conteneur indiqué à l'étape précédente pour obtenir les détails de l'instance de conteneur. Remplacez `MyContainerInstanceID` par l'ID.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   $ aws ecs describe-container-instances \
        ----container-instances MyContainerInstanceID
   ```

   Veuillez noter que la sortie est très longue.

3. Vérifiez que la liste `attributes` contient un objet avec la clé appelée `name` et une valeur `ecs.capability.gmsa-domainless`. Voici un exemple de l'objet.

   Sortie :

   ```
   {
       "name": "ecs.capability.gmsa-domainless"
   }
   ```

Étape 12 : exécuter une tâche Windows

Exécutez une tâche Amazon ECS. S'il n'y a qu'une seule instance de conteneur dans le cluster, vous pouvez utiliser `run-task`. S'il existe de nombreuses instances de conteneur différentes, il peut être plus facile d'utiliser `start-task` et de spécifier l'ID d'instance de conteneur sur lequel exécuter la tâche, plutôt que d'ajouter des contraintes de placement à la définition de la tâche pour contrôler le type d'instance de conteneur sur lequel exécuter cette tâche.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

   ```
   aws ecs run-task --task-definition windows-gmsa-domainless-task \
       --enable-execute-command --cluster windows-domainless-gmsa-cluster
   ```

   Notez l'ID de tâche qui est renvoyé par la commande.
2. Exécutez la commande suivante pour vérifier que la tâche a démarré. Cette commande attend et ne renvoie pas l'invite du shell tant que la tâche ne démarre pas. Remplacez `MyTaskID` par l'ID de tâche de l'étape précédente.

   ```
   $ aws ecs wait tasks-running --task MyTaskID
   ```

Étape 13 : Vérifiez que le conteneur contient gMSA informations d’identification

Vérifiez que le conteneur de la tâche possède un jeton Kerberos. gMSA

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws ecs execute-command \
--task MyTaskID \
--container windows_sample_app \
--interactive \
--command powershell.exe
```

Le résultat sera une PowerShell invite.

2. Exécutez la commande suivante dans le PowerShell terminal à l'intérieur du conteneur.

```
PS C:\> klist get ExampleAccount$
```

Dans la sortie, notez que le `Principal` est celui que vous avez créé précédemment.

Étape 14 : nettoyer

Une fois que vous avez terminé ce didacticiel, vous devez nettoyer les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources inutilisées.

Cette étape utilise le AWS CLI. Vous pouvez exécuter ces commandes dans AWS CloudShell dans le shell par défaut, qui est `bash`.

1. Arrêtez la tâche. Remplacez `MyTaskID` par l'ID de tâche de l'étape 12, Étape 12 : exécuter une tâche Windows.

```
$ aws ecs stop-task --task MyTaskID
```

2. Résiliez l'instance Amazon EC2. Ensuite, l'instance de conteneur du cluster sera automatiquement supprimée au bout d'une heure.

   Vous pouvez détecter et résilier l'instance à l'aide de la console Amazon EC2. Vous pouvez également exécuter la commande suivante. Pour exécuter la commande, recherchez l'ID d'instance EC2 dans le résultat de la commande `aws ecs describe-container-instances` de l'étape 1, Étape 11 : vérifier l'instance de conteneur. i-10a64379 est un exemple d'ID d'instance EC2.

```
$ aws ec2 terminate-instances --instance-ids MyInstanceID
```

3. Supprimez le fichier CredSpec dans Amazon S3. Remplacez `MyBucket` par le nom de votre compartiment Simple Storage Service (Amazon S3).

```
$ aws s3api delete-object --bucket MyBucket --key ecs-domainless-gmsa-credspec/
gmsa-cred-spec.json
```

4. Supprimez le secret dans Secrets Manager. Si vous avez plutôt utilisé SSM Parameter Store, supprimez le paramètre.

   La commande suivante utilise des barres obliques inverses qui sont utilisées par `sh` et les shells compatibles. Cette commande n'est pas compatible avec PowerShell. Vous devez modifier la commande pour l'utiliser avec PowerShell.

```
$ aws secretsmanager delete-secret --secret-id gmsa-plugin-input \
    --force-delete-without-recovery
```

5. Annulez l'enregistrement et supprimez la définition de tâche. En annulant l'enregistrement de la définition de tâche, vous la marquez comme inactive afin qu'elle ne puisse pas être utilisée pour démarrer de nouvelles tâches. Ensuite, vous pouvez supprimer la définition de tâche.

   a. Annulez l'enregistrement de la définition de tâche en spécifiant la version. ECS crée automatiquement des versions des définitions de tâches, qui sont numérotées à partir de 1. Vous faites référence aux versions dans le même format que les étiquettes sur les images des conteneurs, telles que `:1`.

```
$ aws ecs deregister-task-definition --task-definition windows-gmsa-domainless-
task:1
```

   b. Supprimez la définition de tâche.

```
$ aws ecs delete-task-definitions --task-definition windows-gmsa-domainless-
task:1
```

6. (Facultatif) Supprimez le cluster ECS, si vous en avez créé un.

```
$ aws ecs delete-cluster --cluster windows-domainless-gmsa-cluster
```

Débogage d'Amazon ECS sans domaine gMSA pour conteneurs Windows

Statut de tâche Amazon ECS

ECS essaie de démarrer une tâche une seule fois. Toute tâche présentant un problème est arrêtée et définie sur le statut `STOPPED`. Il existe deux types courants de problèmes liés aux tâches. Premièrement, les tâches qui n'ont pas pu être démarrées. Deuxièmement, les tâches pour lesquelles l'application s'est arrêtée dans l'un des conteneurs. Dans le Console de gestion AWS champ Motif de l'arrêt de la tâche, recherchez la raison pour laquelle la tâche a été arrêtée. Dans l' AWS CLI, décrivez la tâche et examinez la `stoppedReason`. Pour les étapes à suivre dans Console de gestion AWS le AWS CLI sable, voirAffichage des erreurs liées aux tâches Amazon ECS arrêtées.

Événements Windows

Les événements Windows pour gMSA dans les conteneurs sont enregistrés dans le fichier journal `Microsoft-Windows-Containers-CCG` et se trouvent dans l'Observateur d'événements, dans la section Applications et services de `Logs\Microsoft\Windows\Containers-CCG \Admin`. Pour obtenir d'autres conseils de débogage, veuillez consulter Résoudre des problèmes de comptes de service administré de groupe pour des conteneurs Windows sur le site Web de Microsoft Learn.

Plug-in gMSA d'agent ECS

La journalisation du plug-in gMSA pour l'agent ECS sur l'instance de conteneur Windows se trouve dans le répertoire suivant, `C:/ProgramData/Amazon/gmsa-plugin/`. Consultez ce journal pour voir si les informations d'identification de l'utilisateur sans domaine ont été téléchargées depuis l'emplacement de stockage, tel que Secrets Manager, et si le format des informations d'identification a été lu correctement.

Utilisation des gMSA pour les conteneurs Windows EC2 pour Amazon ECS

Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Windows par l'intermédiaire d'un type de compte de service spécial appelé gMSA (group Managed Service Account).

Les applications réseau Windows comme les applications .NET font souvent appel à Active Directory pour faciliter la gestion de l'authentification et des autorisations entre les utilisateurs et les services. Les développeurs conçoivent généralement leurs applications de façon qu'elles s'intègrent à Active Directory et s'exécutent à cette fin sur des serveurs joints à un domaine. Étant donné que les

conteneurs Windows ne peuvent pas être joints à un domaine, vous devez configurer un conteneur Windows pour qu'il s'exécute avec un compte gMSA.

Un conteneur Windows s'exécutant avec un compte gMSA attend de son instance Amazon EC2 hôte qu'elle récupère les informations d'identification gMSA auprès du contrôleur de domaine Active Directory et qu'elle les communique à l'instance de conteneur. Pour plus d'informations, consultez Création de groupes gMSA pour des conteneurs Windows.

> **Note**
>
> Cette fonction n'est pas prise en charge par les conteneurs Windows sur Fargate.

Rubriques

- Considérations
- Conditions préalables
- Configuration gMSA pour les conteneurs Windows sur Amazon ECS

Considérations

Les points suivants doivent être pris en compte lorsqu'il s'agit d'utiliser des groupes gMSA pour des conteneurs Windows :

- Lorsque vous utilisez l'AMI complète Amazon ECS-optimized Windows Server 2016 pour vos instances de conteneur, le nom d'hôte du conteneur doit être identique au nom de compte GMSA défini dans le fichier de spécifications d'identification. Pour spécifier le nom d'hôte d'un conteneur, utilisez le paramètre de définition de conteneur `hostname`. Pour de plus amples informations, veuillez consulter Paramètres réseau.
- Vous avez choisi entre les gMSA sans domaine et joindre chaque instance à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

  Choisissez ensuite le stockage de données pour CredSpec et, éventuellement, pour les informations d'identification utilisateur Active Directory pour les gMSA sans domaine.

  Amazon ECS utilise un fichier de spécification des informations d'identification Active Directory (CredSpec). Ce fichier contient les métadonnées gMSA servant à propager le contexte du compte

gMSA au conteneur. Vous générez le fichier CredSpec, puis vous le stockez dans l'une des options de stockage CredSpec du tableau suivant, spécifique au système d'exploitation des instances de conteneur. Pour utiliser la méthode sans domaine, une section facultative du fichier CredSpec peut spécifier les informations d'identification dans l'une des options de stockage domainless user credentials du tableau suivant, spécifiques au système d'exploitation des instances de conteneur.

Emplacement de stockage	Linux	Windows
Amazon Simple Storage Service	CredSpec	CredSpec
AWS Secrets Manager	informations d'identification utilisateur sans domaine	informations d'identification utilisateur sans domaine
Amazon EC2 Systems Manager Parameter Store	CredSpec	CredSpec, informations d'identification utilisateur sans domaine
Fichier local	N/A	CredSpec

Conditions préalables

Avant d'utiliser la fonctionnalité gMSA pour les conteneurs Windows avec Amazon ECS, assurez-vous de remplir les conditions suivantes :

- Vous configurez un domaine Active Directory avec les ressources auxquelles vous souhaitez que vos conteneurs accèdent. Amazon ECS prend en charge les configurations suivantes :
  - Un Directory Service Active Directory. Directory Service est un Active Directory AWS géré hébergé sur Amazon EC2. Pour plus d'informations, consultez Getting Started with AWS Managed Microsoft AD dans le Guide d'AWS Directory Service administration.
  - Un répertoire Active Directory sur site. Vous devez vous assurer que l'instance de conteneur Linux Amazon ECS peut se joindre au domaine. Pour de plus amples informations, veuillez consulter AWS Direct Connect.
- Vous disposez d'un compte gMSA existant dans l'Active Directory. Pour plus d'informations, consultez Création de groupes gMSA pour des conteneurs Windows.

- Vous avez choisi d'utiliser gMSA sans domaine ou l'instance de conteneur Amazon ECS Windows hébergeant la tâche Amazon ECS doit être jointe au domaine du répertoire Active Directory et être membre du groupe de sécurité Active Directory qui a accès au compte gMSA.

  En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.
- Vous avez ajouté les autorisations IAM requises. Les autorisations requises dépendent des méthodes que vous choisissez pour les informations d'identification initiales et pour le stockage de la spécification des informations d'identification :
  - Si vous utilisez domainless gMSA pour les informations d'identification initiales, les autorisations IAM pour AWS Secrets Manager sont requises sur le rôle d'instance Amazon EC2.
  - Si vous stockez la spécification des informations d'identification dans SSM Parameter Store, les autorisations IAM d'Amazon EC2 Systems Manager Parameter Store sont requises pour le rôle d'exécution de la tâche.
  - Si vous stockez la spécification des informations d'identification dans Amazon S3, les autorisations IAM d'Amazon Simple Storage Service sont requises sur le rôle d'exécution de la tâche.

Configuration gMSA pour les conteneurs Windows sur Amazon ECS

Pour configurer gMSA pour les conteneurs Windows sur Amazon ECS, vous pouvez suivre le didacticiel complet qui inclut la configuration des prérequis Utilisation de conteneurs Windows Amazon ECS avec un environnement sans domaine gMSA en utilisant le AWS CLI.

Les sections suivantes décrivent la configuration de CredSpec en détail.

Rubriques

- Exemple CredSpec
- Sans domaine gMSA configuration
- Référencement d'un fichier de spécification d'informations d'identification dans une définition de tâche

Exemple CredSpec

Amazon ECS utilise un fichier de spécification d'informations d'identification qui contient les métadonnées gMSA servant à propager le contexte du compte gMSA au conteneur Windows. Vous pouvez générer le fichier de spécification d'informations d'identification et le référencer dans le champ `credentialSpec` de votre définition de tâche. Le fichier de spécification d'informations d'identification ne contient pas de secrets.

Voici un exemple de fichier de spécification d'informations d'identification :

```
{
    "CmsPlugins": [
        "ActiveDirectory"
    ],
    "DomainJoinConfig": {
        "Sid": "S-1-5-21-2554468230-2647958158-2204241789",
        "MachineAccountName": "WebApp01",
        "Guid": "8665abd4-e947-4dd0-9a51-f8254943c90b",
        "DnsTreeName": "contoso.com",
        "DnsName": "contoso.com",
        "NetBiosName": "contoso"
    },
    "ActiveDirectoryConfig": {
        "GroupManagedServiceAccounts": [
            {
                "Name": "WebApp01",
                "Scope": "contoso.com"
            }
        ]
    }
}
```

Sans domaine gMSA configuration

Nous recommandons gMSA sans domaine au lieu de joindre les instances de conteneur à un seul domaine. En utilisant les gMSA sans domaine, l'instance de conteneur n'est pas jointe au domaine, les autres applications de l'instance ne peuvent pas utiliser les informations d'identification pour accéder au domaine et les tâches qui joignent différents domaines peuvent s'exécuter sur la même instance.

1. Avant de charger CredSpec vers l'une des options de stockage, ajoutez des informations au CredSpec à l'aide de l'ARN du secret dans Secrets Manager ou SSM Parameter Store. Pour plus d'informations, veuillez consulter Configuration de spécifications d'informations d'identification supplémentaires pour un cas d'usage avec un hôte de conteneur non joint à un domaine sur le site Web de Microsoft Learn.

   Sans domaine gMSA format des informations d'identification

   Voici le format JSON pour les informations d'identification gMSA sans domaine pour votre Active Directory. Stockez les informations d'identification dans Secrets Manager ou SSM Parameter Store.

   ```
   {
       "username":"WebApp01",
       "password":"Test123!",
       "domainName":"contoso.com"
   }
   ```

2. Ajoutez les informations suivantes au fichier CredSpec contenu dans le `ActiveDirectoryConfig`. Remplacez l'ARN par le secret dans Secrets Manager ou SSM Parameter Store.

   Veuillez noter que la valeur `PluginGUID` doit correspondre au GUID de l'extrait de code suivant et qu'elle est obligatoire.

   ```
       "HostAccountConfig": {
           "PortableCcgVersion": "1",
           "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
           "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
   region:111122223333:secret:gmsa-plugin-input\"}"
       }
   ```

   Vous pouvez également utiliser un secret dans SSM Parameter Store en utilisant l'ARN au format suivant : `\"arn:aws:ssm:aws-region:111122223333:parameter/gmsa-plugin-input\"`.

3. Une fois le fichier CredSpec modifié, il doit ressembler à l'exemple suivant :

   ```
   {
     "CmsPlugins": [
   ```

```
    "ActiveDirectory"
  ],
  "DomainJoinConfig": {
    "Sid": "S-1-5-21-4066351383-705263209-1606769140",
    "MachineAccountName": "WebApp01",
    "Guid": "ac822f13-583e-49f7-aa7b-284f9a8c97b6",
    "DnsTreeName": "contoso",
    "DnsName": "contoso",
    "NetBiosName": "contoso"
  },
  "ActiveDirectoryConfig": {
    "GroupManagedServiceAccounts": [
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      },
      {
        "Name": "WebApp01",
        "Scope": "contoso"
      }
    ],
    "HostAccountConfig": {
      "PortableCcgVersion": "1",
      "PluginGUID": "{859E1386-BDB4-49E8-85C7-3070B13920E1}",
      "PluginInput": "{\"credentialArn\": \"arn:aws:secretsmanager:aws-
region:111122223333:secret:gmsa-plugin-input\"}"
    }
  }
}
```

Référencement d'un fichier de spécification d'informations d'identification dans une définition de tâche

Amazon ECS permet de référencer le chemin du fichier dans le champ `credentialSpecs` d'une définition de tâche. Pour chacune de ces options, vous pouvez fournir `credentialspec:` ou `domainlesscredentialspec:`, selon que vous joignez les instances de conteneur à un seul domaine, ou que vous utilisez gMSA sans domaine, respectivement.

Compartiment Amazon S3

Ajoutez la spécification d'informations d'identification à un compartiment Amazon S3, puis référencez l'Amazon Resource Name (ARN) du compartiment Amazon S3 dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspecdomainless:arn:aws:s3:::${BucketName}/${ObjectName}"
            ],
            ...
        }
    ],
    ...
}
```

Vous devez également ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS pour permettre à vos tâches d'accéder au compartiment Amazon S3.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Sid": "VisualEditor",
            "Effect": "Allow",
            "Action": [
                "s3:Get*",
                "s3:List*"
            ],
            "Resource": [
                "arn:aws:s3:::{bucket_name}",
                "arn:aws:s3:::{bucket_name}/{object}"
            ]
```

```
        }
    ]
}
```

Paramètre SSM Parameter Store

Ajoutez la spécification d'informations d'identification à un compartiment SSM Parameter Store, puis référencez l'Amazon Resource Name (ARN) du paramètre SSM Parameter Store dans le champ `credentialSpecs` de la définition de tâche.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [

 "credentialspecdomainless:arn:aws:ssm:region:111122223333:parameter/parameter_name"
            ],
            ...
        }
    ],
    ...
}
```

Vous devez également ajouter les autorisations suivantes sous forme de stratégie en ligne au rôle IAM d'exécution de tâche Amazon ECS pour permettre à vos tâches d'accéder au paramètre SSM Parameter Store.

JSON

```
{
    "Version":"2012-10-17",
    "Statement": [
        {
            "Effect": "Allow",
            "Action": [
                "ssm:GetParameters"
            ],
```

```
            "Resource": "arn:aws:ssm:*:111122223333:parameter/example*"
        }
    ]
}
```

Fichier local

Référencez le chemin d'un fichier local qui contient les détails de la spécification d'informations d'identification dans le champ `credentialSpecs` de la définition de tâche. Le chemin de fichier référencé doit être relatif au répertoire `C:\ProgramData\Docker\CredentialSpecs` et utiliser la barre oblique inverse (« \ ») comme séparateur de chemin de fichier.

```
{
    "family": "",
    "executionRoleArn": "",
    "containerDefinitions": [
        {
            "name": "",
            ...
            "credentialSpecs": [
                "credentialspec:file://CredentialSpecDir\CredentialSpecFile.json"
            ],
            ...
        }
    ],
    ...
}
```

Utilisation d'EC2 Image Builder pour créer des AMI Amazon personnalisées ECS-optimized

AWS vous recommande d'utiliser les ECS-optimized AMI Amazon car elles sont préconfigurées avec les exigences et les recommandations pour exécuter vos charges de travail de conteneur. Il peut arriver que vous deviez personnaliser votre AMI pour ajouter des logiciels supplémentaires. Vous pouvez utiliser EC2 Image Builder pour la création, la gestion et le déploiement de vos images de serveur. Vous restez propriétaire des images personnalisées créées dans votre compte. Vous pouvez utiliser les pipelines EC2 Image Builder pour automatiser les mises à jour et les correctifs système pour vos images, ou utiliser une commande autonome pour créer une image avec les ressources de configuration que vous avez définies.

Vous créez une recette pour votre image. La recette inclut une image parent et tous les composants supplémentaires. Vous créez également un pipeline qui distribue votre AMI personnalisée.

Vous créez une recette pour votre image. Une recette Image Builder est un document qui définit l'image de base et les composants à appliquer à l'image de base pour produire la configuration souhaitée pour l'image AMI en sortie. Vous créez également un pipeline qui distribue votre AMI personnalisée. Pour plus d'informations, consultez la section Fonctionnement d'EC2 Image Builder dans le Guide de l'utilisateur d'EC2 Image Builder.

Nous vous recommandons d'utiliser l'une des ECS-optimized AMI Amazon suivantes comme « image parent » dans EC2 Image Builder :

- Linux
  - Amazon ECS-optimized AL2023 x86
  - AMI ECS-optimized Amazon Amazon Linux 2023 (arm64)
  - Amazon ECS-optimized Amazon Linux 2, noyau 5 (AMI)
  - AMI ECS-optimized Amazon Linux 2 x86
- Windows
  - Amazon ECS-optimized Windows 2022 complet x86
  - Amazon ECS-optimized Windows 2022 Core x86
  - Amazon ECS-optimized Windows 2019 complet x86
  - Amazon ECS-optimized Windows 2019 Core x86
  - Amazon ECS-optimized Windows 2016 complet x86

Nous vous recommandons également de sélectionner « Utiliser la dernière version disponible du système d'exploitation ». Le pipeline utilisera la gestion des versions sémantique pour l'image parent, ce qui permet de détecter les mises à jour des dépendances dans les tâches planifiées automatiquement. Pour plus d'informations, consultez la section Gestion des versions sémantique dans le Guide de l'utilisateur EC2 Image Builder.

AWS met régulièrement à jour les images Amazon ECS-optimized AMI avec des correctifs de sécurité et la nouvelle version de l'agent de conteneur. Lorsque vous utilisez un ID AMI comme image parent dans votre recette d'image, vous devez vérifier régulièrement les mises à jour de l'image parent. S'il y a des mises à jour, vous devez créer une nouvelle version de votre recette avec l'AMI mise à jour. Cela garantit que vos images personnalisées incorporent la dernière version de l'image parent. Pour plus d'informations sur la création d'un flux de travail pour mettre à jour

automatiquement vos instances EC2 dans votre cluster Amazon ECS avec les nouvelles AMI, consultez la section Comment créer un pipeline de renforcement des AMI et automatiser les mises à jour de votre flotte d'instances ECS.

Vous pouvez également spécifier le Amazon Resource Name (ARN) d'une image parent publiée via un pipeline EC2 Image Builder géré. Amazon publie régulièrement des images Amazon ECS-optimized AMI via des pipelines gérés. Ces images sont publiquement accessibles. Vous devez disposer des autorisations nécessaires pour accéder à l'image. Lorsque vous utilisez un ARN d'image au lieu d'une AMI dans votre recette Image Builder, votre pipeline utilise automatiquement la version la plus récente de l'image parent à chaque exécution. Cette approche élimine le besoin de créer manuellement de nouvelles versions de recettes pour chaque mise à jour.

Nettoyez l' ECS-optimized AMI Amazon

Lorsque vous utilisez une ECS-optimized AMI Amazon comme image parent, vous devez nettoyer l'image pour éviter les problèmes transitoires. L' ECS-optimized AMI Amazon est préconfigurée pour que l'agent Amazon ECS démarre et s'enregistre automatiquement en tant qu'instance de conteneur auprès d'Amazon ECS. Son utilisation comme image de base sans nettoyage approprié peut entraîner des problèmes dans votre AMI personnalisée.

Pour nettoyer l'image en vue d'une utilisation future, créez un composant qui exécute les commandes suivantes pour arrêter le package ecs-init et les processus Docker :

```
sudo systemctl stop ecs
sudo systemctl stop docker
```

Supprimez tous les fichiers journaux de l'instance actuelle pour éviter de les conserver lors de l'enregistrement de l'image. Utilisez l'exemple de script de la section Pratiques exemplaires en matière de sécurité pour EC2 Image Builder pour nettoyer les différents fichiers de l'instance.

Pour nettoyer les données spécifiques d'Amazon ECS, exécutez les commandes suivantes :

```
sudo rm -rf /var/log/ecs/*
sudo rm /var/lib/ecs/data/agent.db
```

Pour plus d'informations sur la création d' ECS-optimized AMI Amazon personnalisées, consultez Comment créer une AMI personnalisée à partir d'une ECS-optimized AMI Amazon ? dans le AWS Knowledge Center.

Utilisation de l'ARN d'image avec l'infrastructure en tant que code (IaC)

Vous pouvez configurer la recette à l'aide de la console EC2 Image Builder, de l'infrastructure sous forme de code (par exemple CloudFormation) ou AWS du SDK. Lorsque vous spécifiez une image parent dans votre recette, vous pouvez spécifier un ID d'AMI EC2, un ARN d'image Image Builder, un identifiant de AWS Marketplace produit ou une image de conteneur. AWS publie publiquement à la fois les ID AMI et les ARN des images Image Builder des ECS-Optimized AMI Amazon. Le format de l'ARN d'image est le suivant :

```
arn:${Partition}:imagebuilder:${Region}:${Account}:image/${ImageName}/${ImageVersion}
```

L'`ImageVersion` a le format suivant. Remplacez*`major`*, *`minor`* et *`patch`* par les dernières valeurs.

```
<major>.<minor>.<patch>
```

Vous pouvez remplacer `major`, `minor` et `patch` par des valeurs spécifiques ou utiliser l'ARN sans version d'une image, afin que votre pipeline reste à jour avec la dernière version de l'image parent. Un ARN sans version utilise le format générique « x.x.x » pour représenter la version de l'image. Cette approche permet au service Image Builder de résoudre automatiquement la version la plus récente de l'image. L'utilisation d'un ARN sans version garantit que votre référence pointe toujours vers la dernière image disponible, ce qui rationalise le processus de mise à jour des images dans votre déploiement. Lorsque vous créez une recette avec la console, EC2 Image Builder identifie automatiquement l'ARN de votre image parent. Lorsque vous utilisez l'IaC pour créer la recette, vous devez identifier l'ARN et sélectionner la version d'image souhaitée ou utiliser un ARN sans version pour indiquer la dernière image disponible. Nous vous recommandons de créer un script automatique pour filtrer et n'afficher que les images correspondant à vos critères. Le script Python suivant montre comment récupérer une liste d' ECS-optimized AMI Amazon.

Le script accepte deux arguments facultatifs : `owner` et `platform`, avec les valeurs par défaut « Amazon » et « Windows » respectivement. Les valeurs valides pour l'argument du propriétaire sont : `Self`, `Shared`, `Amazon` et `ThirdParty`. Les valeurs valides pour l'argument de plateforme sont `Windows` et `Linux`. Par exemple, si vous exécutez le script avec les `owner` arguments set to `Amazon` et `platform` set to`Linux`, le script génère une liste d'images publiées par Amazon, y compris les ECS-Optimized images Amazon.

```
import boto3
import argparse
```

```
def list_images(owner, platform):
    # Create a Boto3 session
    session = boto3.Session()

    # Create an EC2 Image Builder client
    client = session.client('imagebuilder')

    # Define the initial request parameters
    request_params = {
        'owner': owner,
        'filters': [
            {
                'name': 'platform',
                'values': [platform]
            }
        ]
    }

    # Initialize the results list
    all_images = []

    # Get the initial response with the first page of results
    response = client.list_images(**request_params)

    # Extract images from the response
    all_images.extend(response['imageVersionList'])

    # While 'nextToken' is present, continue paginating
    while 'nextToken' in response and response['nextToken']:
        # Update the token for the next request
        request_params['nextToken'] = response['nextToken']

        # Get the next page of results
        response = client.list_images(**request_params)

        # Extract images from the response
        all_images.extend(response['imageVersionList'])

    return all_images

def main():
    # Initialize the parser
    parser = argparse.ArgumentParser(description="List AWS images based on owner and
 platform")
```

```
    # Add the parameters/arguments
    parser.add_argument("--owner", default="Amazon", help="The owner of the images.
 Default is 'Amazon'.")
    parser.add_argument("--platform", default="Windows", help="The platform type of the
 images. Default is 'Windows'.")

    # Parse the arguments
    args = parser.parse_args()

    # Retrieve all images based on the provided owner and platform
    images = list_images(args.owner, args.platform)

    # Print the details of the images
    for image in images:
        print(f"Name: {image['name']}, Version: {image['version']}, ARN:
 {image['arn']}")

if __name__ == "__main__":
    main()
```

Utilisation de l'ARN de l'image avec CloudFormation

Une recette d’image Image Builder est un plan qui spécifie l’image parent et les composants nécessaires pour obtenir la configuration prévue de l’image de sortie. Vous utilisez la ressource `AWS::ImageBuilder::ImageRecipe`. Définissez la valeur `ParentImage` sur l’ARN d’image. Utilisez l’ARN sans version de l’image souhaitée pour vous assurer que votre pipeline utilise toujours la version la plus récente de l’image. Par exemple, `arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-optimized-x86/x.x.x`. La définition de `AWS::ImageBuilder::ImageRecipe` ressource suivante utilise un ARN d’image géré par Amazon :

```
ECSRecipe:
    Type: AWS::ImageBuilder::ImageRecipe
    Properties:
      Name: MyRecipe
      Version: '1.0.0'
      Components:
        - ComponentArn: [<The component arns of the image recipe>]
      ParentImage: "arn:aws:imagebuilder:us-east-1:aws:image/amazon-linux-2023-ecs-
optimized-x86/x.x.x"
```

Pour plus d'informations sur la ressource `AWS::ImageBuilder::ImageRecipe`, consultez le Guide de l'utilisateur AWS CloudFormation .

Vous pouvez automatiser la création de nouvelles images dans votre pipeline en définissant la propriété `Schedule` de la ressource `AWS::ImageBuilder::ImagePipeline`. Le calendrier inclut une condition de départ et une expression cron. Pour plus d'informations, consultez `AWS::ImageBuilder::ImagePipeline` dans le Guide de l'utilisateur AWS CloudFormation .

Dans l'exemple `AWS::ImageBuilder::ImagePipeline` suivant, le pipeline exécute une construction à 10 h 00, heure universelle coordonnée (UTC) tous les jours. Définissez les valeurs `Schedule` suivantes :

- Définissez `PipelineExecutionStartCondition` sur `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. Le build ne démarre que si des ressources dépendantes telles que l'image parent ou les composants, qui utilisent le caractère générique « x » dans leurs versions sémantiques, sont mises à jour. Cela garantit que la version intègre les dernières mises à jour de ces ressources.
- ScheduleExpression Défini sur l'expression `(0 10 * * ? *)` cron.

```
ECSPipeline:
    Type: AWS::ImageBuilder::ImagePipeline
    Properties:
      Name: my-pipeline
      ImageRecipeArn: <arn of the recipe you created in previous step>
      InfrastructureConfigurationArn: <ARN of the infrastructure configuration
 associated with this image pipeline>
      Schedule:
        PipelineExecutionStartCondition:
 EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE
        ScheduleExpression: 'cron(0 10 * * ? *)'
```

Utilisation de l'ARN d'image avec Terraform

L'approche pour spécifier l'image parent et le calendrier de votre pipeline dans Terraform s'aligne sur celle de AWS CloudFormation. Vous utilisez la ressource `aws_imagebuilder_image_recipe`. Définissez la valeur `parent_image` sur l'ARN d'image. Utilisez l'ARN sans version de l'image

souhaitée pour vous assurer que votre pipeline utilise toujours la version la plus récente de l'image. Pour plus d'informations, consultez la section `aws_imagebuilder_image_recipe` dans la documentation Terraform.

Dans le bloc de configuration du calendrier de la `aws_imagebuilder_image_pipeline resource`, définissez la valeur de l'argument `schedule_expression` sur une expression cron de votre choix pour spécifier la fréquence d'exécution du pipeline, puis définissez la valeur de `pipeline_execution_start_condition` sur `EXPRESSION_MATCH_AND_DEPENDENCY_UPDATES_AVAILABLE`. Pour plus d'informations, consultez la section `aws_imagebuilder_image_pipeline` dans la documentation Terraform.

Utilisation AWS Deep Learning Containers sur Amazon ECS

AWS Les Deep Learning Containers fournissent un ensemble d'images Docker pour la formation et le service de modèles dans TensorFlow et Apache MXNet (incubation) sur Amazon ECS. Les Deep Learning Containers permettent d'optimiser les environnements grâce TensorFlow aux bibliothèques NVIDIA CUDA (pour les instances GPU) et Intel MKL (pour les instances CPU). Les images de conteneur pour les conteneurs Deep Learning Containers sont disponibles dans Amazon ECR pour être référencés dans les définitions de tâches Amazon ECS. Vous pouvez utiliser les conteneurs Deep Learning Containers avec Amazon Elastic Inference pour réduire vos coûts d'inférence.

Pour commencer à utiliser les conteneurs Deep Learning sans Elastic Inference sur Amazon ECS, consultez la section Conteneurs Deep Learning sur Amazon ECS dans le Guide du développeur AWS Apprentissage profond (deep learning) AMIs .

Tutoriel : Création d'un service à l'aide d'un Blue/Green déploiement

Amazon ECS a intégré blue/green les déploiements dans l'assistant de création de service sur la console Amazon ECS. Pour de plus amples informations, veuillez consulter Création d'un déploiement de mise à jour propagée Amazon ECS.

Le didacticiel suivant explique comment créer un service Amazon ECS contenant une tâche Fargate qui utilise blue/green le type de déploiement avec le. AWS CLI

> **Note**
>
> Support pour effectuer un blue/green déploiement a été ajouté pour CloudFormation. Pour plus d'informations, consultez la section Exécuter des blue/green déploiements Amazon ECS CodeDeploy via CloudFormation le guide de l'AWS CloudFormation utilisateur.

Conditions préalables

Le didacticiel suppose de remplir les prérequis suivants :

- La dernière version du AWS CLI est installée et configurée. Pour plus d'informations sur l'installation ou la mise à niveau du AWS CLI, voir Installation du AWS Command Line Interface.
- Vous devez avoir suivi les étapes de Configurer l'utilisation d'Amazon ECS.
- Votre utilisateur IAM dispose des autorisations requises spécifiées dans l’exemple de politique IAM Amazon ECS_FullAccess.
- Vous avez un créé un VPC et un groupe de sécurité prêts à être utilisés. Pour de plus amples informations, veuillez consulter the section called “Créer un Virtual Private Cloud”.
- Le rôle Amazon ECS CodeDeploy IAM est créé. Pour de plus amples informations, veuillez consulter Amazon ECS CodeDeploy Rôle IAM.

Étape 1 : Créer un Application Load Balancer

Les services Amazon ECS utilisant le type de blue/green déploiement peuvent utiliser un Application Load Balancer, un Network Load Balancer, un Service Connect ou être headless. Pour gérer le transfert de trafic, vous avez besoin d'un équilibreur de charge ou de Service Connect. Ce didacticiel utilise un équilibreur de charge Application Load Balancer.

Pour créer un Application Load Balancer

1. Utilisez la commande crate-load-balancer pour créer un Application Load Balancer. Spécifiez deux sous-réseaux qui ne proviennent pas de la même zone de disponibilité, ainsi qu'un groupe de sécurité.

```
aws elbv2 create-load-balancer \
     --name bluegreen-alb \
     --subnets subnet-abcd1234 subnet-abcd5678 \
```

```
    --security-groups sg-abcd1234 \
    --region us-east-1
```

Les données de sortie contiennent l'Amazon Resource Name (ARN) de l'équilibreur de charge, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642
```

2. Utilisez la commande create-target-group pour créer un groupe cible. Ce groupe cible achemine le trafic vers la tâche d'origine définie dans votre service.

```
aws elbv2 create-target-group \
    --name bluegreentarget1 \
    --protocol HTTP \
    --port 80 \
    --target-type ip \
    --vpc-id vpc-abcd1234 \
    --region us-east-1
```

Les données de sortie contiennent l'ARN du groupe cible, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4
```

3. Utilisez la commande create-listener pour créer un écouteur d'équilibreur de charge avec une règle par défaut qui transfère les demandes au groupe cible :

```
aws elbv2 create-listener \
    --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
    --protocol HTTP \
    --port 80 \
    --default-actions
 Type=forward,TargetGroupArn=arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup
bluegreentarget1/209a844cd01825a4 \
    --region us-east-1
```

Les données de sortie contiennent l'ARN de l'écouteur, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:listener/app/bluegreen-alb/
e5ba62739c16e642/665750bec1b03bd4
```

Étape 2 : créer un cluster Amazon ECS

Utilisez la commande create-cluster pour créer un cluster nommé `tutorial-bluegreen-cluster` à utiliser.

```
aws ecs create-cluster \
     --cluster-name tutorial-bluegreen-cluster \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du cluster, au format suivant :

```
arn:aws:ecs:region:aws_account_id:cluster/tutorial-bluegreen-cluster
```

Étape 3 : Enregistrer une définition de tâche

Utilisez la commande register-task-definition pour enregistrer une définition de tâche compatible avec Fargate. Cela exige d'utiliser le mode réseau `awsvpc`. Voici l'exemple de définition de tâche utilisé pour ce didacticiel.

Pour commencer, créez un fichier nommé `fargate-task.json` avec le contenu suivant. Assurez-vous d'utiliser l'ARN de votre rôle d'exécution de tâche. Pour de plus amples informations, veuillez consulter Rôle IAM d'exécution de tâche Amazon ECS.

```
{
    "family": "sample-fargate",
    "networkMode": "awsvpc",
    "containerDefinitions": [
        {
            "name": "sample-app",
            "image": "public.ecr.aws/docker/library/httpd:latest",
            "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\""
            ]
        }
    ],
    "requiresCompatibilities": [
        "FARGATE"
    ],
    "cpu": "256",
    "memory": "512"
}
```

Enregistrez ensuite la définition de tâche à l'aide du fichier `fargate-task.json` que vous avez créé.

```
aws ecs register-task-definition \
     --cli-input-json file://fargate-task.json \
     --region us-east-1
```

Étape 4 : créer un service Amazon ECS

Utilisez la commande create-service pour créer un service.

Pour commencer, créez un fichier nommé `service-bluegreen.json` avec le contenu suivant.

```
{
    "cluster": "tutorial-bluegreen-cluster",
    "serviceName": "service-bluegreen",
    "taskDefinition": "tutorial-task-def",
    "loadBalancers": [
        {
```

```
            "targetGroupArn":
 "arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4",
            "containerName": "sample-app",
            "containerPort": 80
        }
    ],
    "launchType": "FARGATE",
    "schedulingStrategy": "REPLICA",
    "deploymentController": {
        "type": "CODE_DEPLOY"
    },
    "platformVersion": "LATEST",
    "networkConfiguration": {
       "awsvpcConfiguration": {
          "assignPublicIp": "ENABLED",
          "securityGroups": [ "sg-abcd1234" ],
          "subnets": [ "subnet-abcd1234", "subnet-abcd5678" ]
       }
    },
    "desiredCount": 1
}
```

Ensuite, créez votre service à l'aide du fichier `service-bluegreen.json` que vous avez créé.

```
aws ecs create-service \
     --cli-input-json file://service-bluegreen.json \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du service, au format suivant :

```
arn:aws:ecs:region:aws_account_id:service/tutorial-bluegreen-cluster/service-bluegreen
```

Étape 5 : Création du AWS CodeDeploy Ressources

Suivez les étapes ci-dessous pour créer votre CodeDeploy application, le groupe cible Application Load Balancer pour le groupe de CodeDeploy déploiement et le groupe de CodeDeploy déploiement.

Pour créer des CodeDeploy ressources

1. Utilisez la commande create-application pour créer une CodeDeploy application. Précisez la plateforme de calcul `ECS`.

```
aws deploy create-application \
     --application-name tutorial-bluegreen-app \
     --compute-platform ECS \
     --region us-east-1
```

Les données de sortie contiennent l'ID d'application, au format suivant :

```
{
    "applicationId": "b8e9c1ef-3048-424e-9174-885d7dc9dc11"
}
```

2. Utilisez la commande create-target-group pour créer un deuxième groupe cible Application Load Balancer, qui sera utilisé lors de la création de votre groupe de déploiement. CodeDeploy

```
aws elbv2 create-target-group \
     --name bluegreentarget2 \
     --protocol HTTP \
     --port 80 \
     --target-type ip \
     --vpc-id "vpc-0b6dd82c67d8012a1" \
     --region us-east-1
```

Les données de sortie contiennent l'ARN du groupe cible, au format suivant :

```
arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc
```

3. Utilisez la commande create-deployment-group pour créer un groupe de déploiement. CodeDeploy

Pour commencer, créez un fichier nommé `tutorial-deployment-group.json` avec le contenu suivant. Cet exemple utilise la ressource que vous avez créée. Pour le`serviceRoleArn`, spécifiez l'ARN de votre rôle Amazon ECS CodeDeploy IAM. Pour de plus amples informations, veuillez consulter Amazon ECS CodeDeploy Rôle IAM.

```
{
   "applicationName": "tutorial-bluegreen-app",
   "autoRollbackConfiguration": {
      "enabled": true,
      "events": [ "DEPLOYMENT_FAILURE" ]
```

```
   },
   "blueGreenDeploymentConfiguration": {
      "deploymentReadyOption": {
         "actionOnTimeout": "CONTINUE_DEPLOYMENT",
         "waitTimeInMinutes": 0
      },
      "terminateBlueInstancesOnDeploymentSuccess": {
         "action": "TERMINATE",
         "terminationWaitTimeInMinutes": 5
      }
   },
   "deploymentGroupName": "tutorial-bluegreen-dg",
   "deploymentStyle": {
      "deploymentOption": "WITH_TRAFFIC_CONTROL",
      "deploymentType": "BLUE_GREEN"
   },
   "loadBalancerInfo": {
      "targetGroupPairInfoList": [
        {
          "targetGroups": [
             {
                 "name": "bluegreentarget1"
             },
             {
                 "name": "bluegreentarget2"
             }
          ],
          "prodTrafficRoute": {
              "listenerArns": [
                  "arn:aws:elasticloadbalancing:region:aws_account_id:listener/
app/bluegreen-alb/e5ba62739c16e642/665750bec1b03bd4"
              ]
          }
        }
      ]
   },
   "serviceRoleArn": "arn:aws:iam::aws_account_id:role/ecsCodeDeployRole",
   "ecsServices": [
       {
           "serviceName": "service-bluegreen",
           "clusterName": "tutorial-bluegreen-cluster"
       }
   ]
```

```
}
```

Créez ensuite le groupe CodeDeploy de déploiement.

```
aws deploy create-deployment-group \
     --cli-input-json file://tutorial-deployment-group.json \
     --region us-east-1
```

Les données de sortie contiennent l'ID du groupe de déploiement, au format suivant :

```
{
    "deploymentGroupId": "6fd9bdc6-dc51-4af5-ba5a-0a4a72431c88"
}
```

Étape 6 : créer et surveiller un CodeDeploy déploiement

Suivez les étapes ci-dessous pour créer et télécharger un fichier de spécification d'application (AppSpec fichier) et un CodeDeploy déploiement.

Pour créer et surveiller un CodeDeploy déploiement

1. Créez et chargez un AppSpec fichier en suivant les étapes ci-dessous.

   a. Créez un fichier nommé `appspec.yaml` avec le contenu du groupe de déploiement CodeDeploy . Cet exemple utilise les ressources que vous avez créées précédemment dans le didacticiel.

```
version: 0.0
Resources:
  - TargetService:
      Type: AWS::ECS::Service
      Properties:
        TaskDefinition: "arn:aws:ecs:region:aws_account_id:task-
definition/first-run-task-definition:7"
        LoadBalancerInfo:
          ContainerName: "sample-app"
          ContainerPort: 80
        PlatformVersion: "LATEST"
```

b. Utilisez la commande s3 mb pour créer un compartiment Amazon S3 pour le AppSpec fichier.

```
aws s3 mb s3://tutorial-bluegreen-bucket
```

c. Utilisez la commande s3 cp pour charger le AppSpec fichier dans le compartiment Amazon S3.

```
aws s3 cp ./appspec.yaml s3://tutorial-bluegreen-bucket/appspec.yaml
```

2. Créez le CodeDeploy déploiement en suivant les étapes ci-dessous.

a. Créez un fichier nommé `create-deployment.json` avec le contenu du CodeDeploy déploiement. Cet exemple utilise les ressources que vous avez créées précédemment dans le didacticiel.

```
{
    "applicationName": "tutorial-bluegreen-app",
    "deploymentGroupName": "tutorial-bluegreen-dg",
    "revision": {
        "revisionType": "S3",
        "s3Location": {
            "bucket": "tutorial-bluegreen-bucket",
            "key": "appspec.yaml",
            "bundleType": "YAML"
        }
    }
}
```

b. Utilisez la commande create-deployment pour créer le déploiement.

```
aws deploy create-deployment \
     --cli-input-json file://create-deployment.json \
     --region us-east-1
```

Les données de sortie contiennent l'ID du déploiement, au format suivant :

```
{
    "deploymentId": "d-RPCR1U3TW"
}
```

c. Utilisez la commande get-deployment-target pour obtenir les détails du déploiement, en spécifiant le code `deploymentId` indiqué dans la sortie précédente.

```
aws deploy get-deployment-target \
     --deployment-id "d-IMJU3A8TW" \
     --target-id tutorial-bluegreen-cluster:service-bluegreen \
     --region us-east-1
```

Continuez de récupérer les détails du déploiement jusqu'à ce que le statut soit `Succeeded`, comme indiqué dans la sortie suivante.

```
{
    "deploymentTarget": {
        "deploymentTargetType": "ECSTarget",
        "ecsTarget": {
            "deploymentId": "d-RPCR1U3TW",
            "targetId": "tutorial-bluegreen-cluster:service-bluegreen",
            "targetArn": "arn:aws:ecs:region:aws_account_id:service/tutorial-
bluegreen-cluster/service-bluegreen",
            "lastUpdatedAt": 1543431490.226,
            "lifecycleEvents": [
                {
                    "lifecycleEventName": "BeforeInstall",
                    "startTime": 1543431361.022,
                    "endTime": 1543431361.433,
                    "status": "Succeeded"
                },
                {
                    "lifecycleEventName": "Install",
                    "startTime": 1543431361.678,
                    "endTime": 1543431485.275,
                    "status": "Succeeded"
                },
                {
                    "lifecycleEventName": "AfterInstall",
                    "startTime": 1543431485.52,
                    "endTime": 1543431486.033,
                    "status": "Succeeded"
                },
                {
                    "lifecycleEventName": "BeforeAllowTraffic",
                    "startTime": 1543431486.838,
```

```
                    "endTime": 1543431487.483,
                    "status": "Succeeded"
                },
                {
                    "lifecycleEventName": "AllowTraffic",
                    "startTime": 1543431487.748,
                    "endTime": 1543431488.488,
                    "status": "Succeeded"
                },
                {
                    "lifecycleEventName": "AfterAllowTraffic",
                    "startTime": 1543431489.152,
                    "endTime": 1543431489.885,
                    "status": "Succeeded"
                }
            ],
            "status": "Succeeded",
            "taskSetsInfo": [
                {
                    "identifer": "ecs-svc/9223370493425779968",
                    "desiredCount": 1,
                    "pendingCount": 0,
                    "runningCount": 1,
                    "status": "ACTIVE",
                    "trafficWeight": 0.0,
                    "targetGroup": {
                        "name": "bluegreentarget1"
                    }
                },
                {
                    "identifer": "ecs-svc/9223370493423413672",
                    "desiredCount": 1,
                    "pendingCount": 0,
                    "runningCount": 1,
                    "status": "PRIMARY",
                    "trafficWeight": 100.0,
                    "targetGroup": {
                        "name": "bluegreentarget2"
                    }
                }
            ]
        }
    }
```

```
}
```

Étape 7 : Nettoyer

Lorsque vous avez terminé ce didacticiel, nettoyez les ressources qui lui sont associées afin d'éviter la facturation de frais pour des ressources que vous n'utilisez pas.

Nettoyage des ressources du didacticiel

1. Utilisez la commande delete-deployment-group pour supprimer le groupe de déploiement. CodeDeploy

```
aws deploy delete-deployment-group \
     --application-name tutorial-bluegreen-app \
     --deployment-group-name tutorial-bluegreen-dg \
     --region us-east-1
```

2. Utilisez la commande delete-application pour supprimer l' CodeDeploy application.

```
aws deploy delete-application \
     --application-name tutorial-bluegreen-app \
     --region us-east-1
```

3. Utilisez la commande delete-service pour supprimer l'Amazon ECS service. Utiliser l'indicateur `--force` vous permet de supprimer un service, même s'il n'a pas été abaissé à 0 tâche.

```
aws ecs delete-service \
     --service arn:aws:ecs:region:aws_account_id:service/tutorial-bluegreen-
cluster/service-bluegreen \
     --force \
     --region us-east-1
```

4. Utilisez la commande delete-cluster pour supprimer le cluster Amazon ECS.

```
aws ecs delete-cluster \
     --cluster tutorial-bluegreen-cluster \
     --region us-east-1
```

5. Utilisez la commande s3 rm pour supprimer le AppSpec fichier du compartiment Amazon S3.

```
aws s3 rm s3://tutorial-bluegreen-bucket/appspec.yaml
```

6. Utilisez la commande s3 rb pour supprimer le compartiment Amazon S3.

```
aws s3 rb s3://tutorial-bluegreen-bucket
```

7. Utilisez la commande delete-load-balancer pour supprimer l'Application Load Balancer.

```
aws elbv2 delete-load-balancer \
     --load-balancer-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:loadbalancer/app/bluegreen-alb/
e5ba62739c16e642 \
     --region us-east-1
```

8. Utilisez la commande delete-target-group pour supprimer les deux groupes cibles Application Load Balancer.

```
aws elbv2 delete-target-group \
     --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget1/209a844cd01825a4 \
     --region us-east-1
```

```
aws elbv2 delete-target-group \
     --target-group-arn
 arn:aws:elasticloadbalancing:region:aws_account_id:targetgroup/
bluegreentarget2/708d384187a3cfdc \
     --region us-east-1
```

Quotas de service Amazon ECS service

Lorsque vous créez votre Compte AWS, nous définissons la valeur par défaut quotas (également appelées limites) sur votre AWS ressources par région. Si vous tentez de dépasser le quota d'une ressource, la demande échoue.

La console Service Quotas est un emplacement central où vous pouvez consulter et gérer vos quotas pour AWS services, et demandez une augmentation de quota pour la plupart des ressources que vous utilisez. Utilisez les informations sur les quotas que nous fournissons pour gérer vos ressources. Prévoyez de demander les augmentations de quota avant le moment où vous en aurez besoin.

Pour plus d'informations sur les quotas Amazon ECS et Fargate, consultez la section Points de terminaison et quotas Amazon ECS dans le Référence générale d'Amazon Web Services. Pour plus d'informations sur la limitation des API dans l'API Amazon ECS, consultez Limitation des demandes pour l'API Amazon ECS.

Les instances gérées Amazon ECS respectent les quotas Amazon EC2. Pour plus d'informations, consultez la section Quotas de service Elastic Amazon EC2 dans le Guide de l'utilisateur Amazon EC2.

Gestion de votre Amazon ECS et AWS Fargate quotas de services dans le Console de gestion AWS

Amazon ECS s'est intégré à Service Quotas, AWS service qui vous permet de consulter et de gérer vos quotas depuis un emplacement central. Pour plus d'informations, veuillez consulter la rubrique Qu'est-ce que Service Quotas ? dans le Guide de l'utilisateur Service Quotas.

Service Quotas facilite la recherche de la valeur de vos quotas de service Amazon ECS service.

Console de gestion AWS

Pour consulter les quotas de service Amazon ECS et Fargate à l'aide du Console de gestion AWS

1. Ouvrez la console Service Quotas sur https://console.aws.amazon.com/servicequotas/.
2. Dans le panneau de navigation, choisissez Services AWS .
3. Dans la liste services (Services AWS), recherchez et sélectionnez Amazon Elastic Container Service (Amazon ECS) ou AWS Fargate.

Dans la liste Service quotas (Quotas de service), vous pouvez voir le nom du Service Quota, la valeur appliquée (le cas échéant), le quota AWS par défaut et si la valeur du quota est réglable.

4. Pour afficher des informations supplémentaires sur un quota de service, notamment la description, choisissez le nom du quota.
5. (Facultatif) Pour demander une augmentation de quota, sélectionnez le quota que vous souhaitez augmenter, sélectionnez Request quota increase (Demander une augmentation de quota), saisissez ou sélectionnez les informations requises, puis sélectionnez Request (Demander).

Pour mieux gérer les quotas de service à l'aide du Console de gestion AWS voir le Guide de l'utilisateur des quotas de service. Pour demander une augmentation de quota, consultez Demande d'augmentation de quota dans le Guide de l'utilisateur Service Quotas.

AWS CLI

Pour consulter les quotas de service Amazon ECS et Fargate à l'aide du AWS CLI

Exécutez la commande suivante pour afficher les quotas Amazon ECS par défaut.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code ecs \
    --output table
```

Exécutez la commande suivante pour afficher les quotas Fargate par défaut.

```
aws service-quotas list-aws-default-service-quotas \
    --query 'Quotas[*].
{Adjustable:Adjustable,Name:QuotaName,Value:Value,Code:QuotaCode}' \
    --service-code fargate \
    --output table
```

Exécutez la commande suivante pour afficher les quotas Fargate appliqués.

```
aws service-quotas list-service-quotas \
    --service-code fargate
```

> **Note**
>
> Amazon ECS ne prend pas en charge les quotas appliqués.

Pour plus d'informations sur l'utilisation des quotas de service, utilisez le AWS CLI, consultez le Quotas de services AWS CLI Référence de commande. Pour demander une augmentation de quota, consultez la commande `request-service-quota-increase` dans la référence des commandes AWS CLI.

Gestion des quotas de service Amazon ECS et des seuils de limitation d'API

Amazon ECS est intégré à plusieurs Services AWS, y compris Elastic Load Balancing, AWS Cloud Map et Amazon EC2. Grâce à cette intégration fine, Amazon ECS inclut plusieurs fonctionnalités telles que l'équilibrage de charge des services, Service Connect, la mise en réseau des tâches et l'autoscaling de cluster. Amazon ECS et les autres Services AWS qu'il intègre à tous les systèmes maintiennent les quotas de service et les limites de débit des API afin de garantir des performances et une utilisation cohérentes. Ces quotas de service empêchent également l'allocation accidentelle de ressources supérieures à celles nécessaires et protègent contre les actions malveillantes susceptibles d'augmenter votre facture.

En vous familiarisant avec vos quotas de service et les AWS Grâce aux limites de débit des API, vous pouvez planifier la mise à l'échelle de vos charges de travail sans vous soucier d'une dégradation inattendue des performances. Pour plus d'informations, consultez la section Limitation des requêtes pour l'API Amazon ECS.

Lors de la mise à l'échelle de vos charges de travail sur Amazon ECS, nous vous recommandons de prendre en compte le quota de service suivant.

- AWS Fargate a des quotas qui limitent le nombre de tâches exécutées simultanément dans chaque Région AWS. Il existe des quotas pour les deux On-Demand et les tâches Fargate Spot sur Amazon ECS. Chaque quota de service inclut également tous les pods Amazon EKS que vous utilisez sur Fargate.
- Pour les tâches exécutées sur des instances Amazon EC2, le nombre maximal d'instances Amazon EC2 que vous pouvez enregistrer pour chaque cluster est de 5 000. Si vous utilisez l'autoscaling de cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, ou

si vous gérez vous-même les instances Amazon EC2 pour votre cluster, ce quota peut devenir un obstacle au déploiement. Si vous avez besoin de plus de capacité, vous pouvez créer d'autres clusters ou demander une augmentation de quota de service.

- Si vous utilisez l'autoscaling de cluster Amazon ECS avec un fournisseur de capacité de groupe Auto Scaling, tenez compte du quota `Tasks in the PROVISIONING state per cluster` lors du dimensionnement de vos services. Ce quota est le nombre maximum de tâches dans l'état `PROVISIONING` pour chaque cluster pour lesquelles les fournisseurs de capacité peuvent augmenter la capacité. Lorsque vous lancez un grand nombre de tâches simultanément, vous pouvez facilement atteindre ce quota. Par exemple, si vous déployez simultanément des dizaines de services, chacun comportant des centaines de tâches. Dans ce cas, le fournisseur de capacité doit lancer de nouvelles instances de conteneur pour placer les tâches lorsque la capacité du cluster est insuffisante. Pendant que le fournisseur de capacité lance des instances Amazon EC2 supplémentaires, le planificateur de service Amazon ECS continuera probablement à lancer des tâches en parallèle. Toutefois, cette activité peut être limitée en raison d'une capacité de cluster insuffisante. Le planificateur de service Amazon ECS met en œuvre une stratégie de ralentissement et de limitation exponentielle pour réessayer de placer des tâches lorsque de nouvelles instances de conteneur sont lancées. Par conséquent, les délais de déploiement ou de réduction horizontale peuvent être plus lents. Pour éviter cette situation, vous pouvez planifier vos déploiements de service selon l'une des méthodes suivantes. Déployez un grand nombre de tâches sans augmenter la capacité du cluster ou conservez une capacité de cluster inutilisée pour le lancement de nouvelles tâches.

En plus de prendre en compte le quota de service Amazon ECS lors de la mise à l'échelle de vos charges de travail, prenez également en compte le quota de service pour les autres Services AWS qui sont intégrés à Amazon ECS.

Elastic Load Balancing

Vous pouvez configurer vos services Amazon ECS pour utiliser Elastic Load Balancing afin de répartir le trafic de manière uniforme entre les tâches. Pour plus d'informations et les pratiques exemplaires recommandées concernant le choix d'un équilibreur de charge, consultez la section Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.

Quotas de service Elastic Load Balancing

Lorsque vous mettez à l'échelle vos charges de travail, tenez compte des quotas du service Elastic Load Balancing suivants. La plupart des quotas de service Elastic Load Balancing sont ajustables, et vous pouvez demander une augmentation dans la console Service Quotas.

Application Load Balancer

Lorsque vous utilisez un Application Load Balancer, selon votre cas d'utilisation, vous devrez peut-être demander une augmentation de quota pour :

- Le quota `Targets per Application Load Balancer`, qui correspond au nombre de cibles derrière votre Application Load Balancer.
- Le quota `Targets per Target Group per Region`, qui correspond au nombre de cibles derrière vos groupes cibles.

Pour plus d'informations, consultez la section Quotas pour vos Application Load Balancer du Guide de l'utilisateur des Application Load Balancer.

Network Load Balancer

Le nombre de cibles que vous pouvez enregistrer auprès d'un Network Load Balancer est soumis à des limites plus strictes. Lorsque vous utilisez un Network Load Balancer, vous souhaiterez souvent activer le support entre zones, ce qui implique des limites de mise à l'échelle supplémentaires sur les `Targets per Availability Zone Per Network Load Balancer`, qui correspond au nombre maximum de cibles par zone de disponibilité pour chaque Network Load Balancer. Pour plus d'informations, consultez la section Quotas pour vos Network Load Balancer dans le Guide de l'utilisateur pour les Network Load Balancer.

Limitation de l'API Elastic Load Balancing

Lorsque vous configurez un service Amazon ECS pour utiliser un équilibreur de charge, les surveillances de l'état du groupe cible doivent réussir avant que le service ne soit considéré comme sain. Pour effectuer ces surveillances de l'état, Amazon ECS invoque les opérations de l'API Elastic Load Balancing en votre nom. Si vous avez configuré un grand nombre de services avec des équilibreurs de charge dans votre compte, vous risquez de ralentir les déploiements de service en raison d'une éventuelle limitation spécifique aux opérations `RegisterTarget`, `DeregisterTarget` et `DescribeTargetHealth` de l'API Elastic Load Balancing. En cas de limitation, des erreurs de limitation apparaissent dans les messages d'événements de votre service Amazon ECS.

Si vous rencontrez AWS Cloud Map Limitation de l'API, vous pouvez contacter Support pour obtenir des conseils sur la façon d'augmenter votre AWS Cloud Map Limites de limitation des API. Pour plus d'informations sur la surveillance et la résolution de ces erreurs de limitation, consultez la section Gestion des problèmes de limitation d'Amazon ECS.

Interfaces réseau Elastic

Lorsque vos tâches utilisent le mode réseau `awsvpc`, Amazon ECS fournit une interface réseau Elastic (ENI) unique pour chaque tâche. Lorsque vos services Amazon ECS utilisent un équilibreur de charge Elastic Load Balancing, ces interfaces réseau sont également enregistrées en tant que cibles pour le groupe cible approprié défini dans le service.

Quotas de service de l'interface réseau Elastic

Lorsque vous exécutez des tâches utilisant le mode réseau `awsvpc`, une interface réseau Elastic unique est associée à chaque tâche. Si ces tâches doivent être atteintes via Internet, attribuez une adresse IP publique à l'interface réseau Elastic pour ces tâches. Lorsque vous mettez vos charges de travail Amazon ECS à l'échelle, tenez compte de ces deux quotas importants :

- Le `Network interfaces per Region` quota qui est le nombre maximum d'interfaces réseau dans un Région AWS pour votre compte.
- Le quota `Elastic IP addresses per Region`, qui correspond au nombre maximum d'adresses IP élastiques dans une Région AWS.

Ces deux quotas de service sont ajustables et vous pouvez demander une augmentation depuis votre console Service Quotas pour ces derniers. Pour plus d'informations, consultez la section Quotas de service Amazon VPC dans le Guide de l'utilisateur Amazon Virtual Private Cloud.

Pour les charges de travail Amazon ECS hébergées sur des instances Amazon EC2, lorsque vous exécutez des tâches en mode réseau `awsvpc`, tenez compte du quota de service `Maximum network interfaces`, qui correspond au nombre maximum d'instances réseau pour chaque instance Amazon EC2. Ce quota limite le nombre de tâches que vous pouvez placer sur une instance. Vous ne pouvez pas ajuster le quota et il n'est pas disponible dans la console Service Quotas. Pour de plus amples informations, consultez la section Adresses IP par interface réseau et par type d'instance dans le Guide de l'utilisateur Amazon EC2.

Bien que vous ne puissiez pas modifier le nombre d'interfaces réseau qui peuvent être attachées à des instances Amazon EC2, vous pouvez utiliser la fonctionnalité de jonction d'interface réseau

Elastic pour augmenter le nombre d'interfaces réseau disponibles. Par exemple, par défaut, une instance `c5.large` peut avoir jusqu'à trois interfaces réseau. L'interface réseau principale de l'instance est considérée comme une interface réseau. Ainsi, vous pouvez attacher deux interfaces réseau supplémentaires à l'instance. Dans la mesure où chaque tâche utilisant le mode réseau `awsvpc` nécessite une interface réseau, vous ne pouvez généralement exécuter que deux tâches sur ce type d'instance. Cela peut entraîner une sous-utilisation de la capacité de votre cluster. Si vous activez la jonction d'interface réseau Elastic, vous pouvez augmenter la densité de l'interface réseau afin de placer un plus grand nombre de tâches sur chaque instance. Lorsque la jonction est activée, une instance `c5.large` peut avoir jusqu'à 12 interfaces réseau. L'instance de conteneur a alors une interface réseau principale et Amazon ECS crée et attache une interface réseau « de jonction » à l'instance de conteneur. Par conséquent, avec cette configuration, vous pouvez exécuter 10 tâches sur l'instance au lieu des 2 tâches par défaut. Pour plus d'informations, consultez Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux.

Limitation de l'API d'interface réseau Elastic

Lorsque vous exécutez des tâches qui utilisent le mode réseau `awsvpc`, Amazon ECS s'appuie sur les API Amazon EC2 suivantes. Chacune de ces API possède des limites d'API différentes. Pour plus d'informations, consultez la section Limitation des requêtes pour l'API Amazon EC2 dans la Référence API Amazon EC2.

- CreateNetworkInterface
- AttachNetworkInterface
- DetachNetworkInterface
- DeleteNetworkInterface
- DescribeNetworkInterfaces
- DescribeVpcs
- DescribeSubnets
- DescribeSecurityGroups
- DescribeInstances

Si les appels d'API Amazon EC2 sont limités pendant les flux de travail d'allocation d'interface réseau Elastic, le planificateur de service Amazon ECS réessaie automatiquement avec des interruptions exponentielles. Ces mises hors service automatiques peuvent parfois retarder le lancement des tâches, ce qui ralentit les vitesses de déploiement. En cas de limitation de l'API, le message

`Operations are being throttled. Will try again later.` apparaît sur vos messages d'événement de service. Si vous respectez régulièrement les limites de l'API Amazon EC2, vous pouvez contacter Support pour obtenir des conseils sur la manière d'augmenter les limites de limitation de votre API. Pour plus d'informations sur la surveillance et la résolution de ces erreurs de limitation, consultez la section Gestion des problèmes de limitation.

AWS Cloud Map

Découverte des services Amazon ECS et utilisation de Service Connect AWS Cloud Map Des API pour gérer les espaces de noms pour vos services Amazon ECS. Si vos services comportent un grand nombre de tâches, tenez compte des recommandations suivantes.

AWS Cloud Map service quotas

Lorsque les services Amazon ECS sont configurés pour utiliser Service Discovery ou Service Connect, `Tasks per service` le quota, qui est le nombre maximum de tâches pour le service, est affecté par la AWS Cloud Map `Instances per service` quota de service qui est le nombre maximum d'instances pour ce service. En particulier, le quota de service AWS Cloud Map réduit le nombre d'instances que vous pouvez exécuter à un maximum de 1 000 instances par service. Vous ne pouvez pas modifier AWS Cloud Map quota. Pour plus d'informations, consultez Quotas de service AWS Cloud Map.

AWS Cloud Map Limitation d'API

Amazon ECS appelle `ListInstances`, `GetInstancesHealthStatus`, `RegisterInstance`, et `DeregisterInstance` AWS Cloud Map Des API en votre nom. Elles aident à la découverte de services et effectuent des surveillances de l'état lorsque vous lancez une tâche. Lorsque plusieurs services utilisant la découverte de services avec un grand nombre de tâches sont déployés en même temps, cela peut entraîner un dépassement de la AWS Cloud Map Limites de limitation des API. Dans ce cas, vous verrez probablement le message suivant : `Operations are being throttled. Will try again later` dans les messages d'événements de votre service Amazon ECS et ralentissement de la vitesse de déploiement et de lancement des tâches. AWS Cloud Map ne documente pas les limites de limitation pour ces API. Si vous rencontrez des problèmes d'étranglement à cause de ceux-ci, vous pouvez contacter Support pour obtenir des conseils sur l'augmentation des limites de limitation de votre API. Pour plus de recommandations sur la surveillance et la résolution de ces erreurs de limitation, consultez la section Gestion des problèmes de limitation d'Amazon ECS.

Utilisation de l'API pour Amazon ECS

En plus de Console de gestion AWS et le AWS Command Line Interface (AWS CLI), Amazon ECS fournit également une API. Vous pouvez utiliser l'API pour automatiser les tâches de gestion des ressources Amazon ECS.

- Pour obtenir la liste des opérations d'API par ressource Amazon ECS, voir Actions effectuées par la ressource Amazon ECS.
- Pour obtenir la liste alphabétique des opérations d'API, consultez Actions.
- Pour obtenir la liste alphabétique des types de données, consultez Types de données.
- Pour consulter la liste des paramètres de requête courants, reportez-vous à la page Paramètres courants.
- Pour obtenir la description des codes d'erreur, consultez Erreurs courantes.

Pour plus d'informations sur le AWS CLI, voir AWS Command Line Interface référence pour Amazon ECS.

Actions effectuées par la ressource Amazon ECS

La section suivante répertorie les actions de l'API par ressource.

Rubriques

- Actions de configuration du compte
- Actions groupées
- Actions des fournisseurs de capacités du cluster
- Actions relatives aux conteneurs
- Actions des agents de conteneurs
- Actions relatives aux instances de conteneurs
- Actions relatives à l'espace de noms
- Actions de service
- Actions de déploiement de services
- Actions de révision des services

- Actions de définition de tâches
- Actions relatives aux tâches
- Actions relatives à l'ensemble de tâches
- Marquer les actions

Actions de configuration du compte

Les actions d'API suivantes sont disponibles pour les paramètres du compte :

- DeleteAccountSetting
- ListAccountSettings
- PutAccountSetting
- PutAccountSettingDefault

Actions groupées

Les actions d'API suivantes sont disponibles pour les clusters :

- CreateCluster
- DeleteCluster
- DescribeClusters
- ListClusters
- UpdateCluster
- UpdateClusterSettings

Actions des fournisseurs de capacités du cluster

Les API suivantes sont disponibles pour les fournisseurs de capacité de cluster :

- CreateCapacityProvider
- DeleteCapacityProvider
- DescribeCapacityProviders
- PutClusterCapacityProviders

- UpdateCapacityProvider

Actions relatives aux conteneurs

Les API suivantes sont disponibles pour les conteneurs :

- ExecuteCommand

Actions des agents de conteneurs

Les API suivantes sont disponibles pour les agents de conteneur :

- UpdateContainerAgent

Actions relatives aux instances de conteneurs

Les API suivantes sont disponibles pour les instances de conteneur :

- DeregisterContainerInstance
- DescribeContainerInstances
- DeleteAttributes
- ListAttributes
- ListContainerInstances
- PutAttributes
- UpdateContainerInstancesState

Actions relatives à l'espace de noms

> **Note**
>
> Les espaces de noms sont AWS Cloud Map ressource. Les espaces de noms sont nécessaires dans Amazon ECS pour utiliser les fonctionnalités ECS Service Connect ou ECS Service Discovery. Amazon ECS demande des espaces de noms à AWS Cloud Map puis affiche vos espaces de noms dans le Console de gestion AWS. Utilisez le AWS Cloud Map console ou API pour des options de configuration supplémentaires.

Les API suivantes sont disponibles pour les espaces de noms :

- CreateCluster

Actions de service

Les API suivantes sont disponibles pour les services :

- CreateService
- DeleteService
- DescribeServices
- ListServices
- UpdateService

Actions de déploiement de services

Les API suivantes sont disponibles pour les déploiements de services :

- DescribeServiceDeployments
- ListServiceDeployments
- StopServiceDeployment

Actions de révision des services

Les API suivantes sont disponibles pour les révisions de service :

- DescribeServiceRevisions

Actions de définition de tâches

Les API suivantes sont disponibles pour les définitions de tâches :

- DeleteTaskDefinitions
- DeregisterTaskDefinition
- DescribeTaskDefinition

- ListTaskDefinitionFamilies
- ListTaskDefinitions
- RegisterTaskDefinition

Actions relatives aux tâches

Les API suivantes sont disponibles pour les tâches :

- DescribeTasks
- GetTaskProtection
- ListTasks
- RunTask
- StartTask
- StopTask
- UpdateTaskProtection

Actions relatives à l'ensemble de tâches

Les API suivantes sont disponibles pour les ensembles de tâches :

- CreateTaskSet
- DeleteTaskSet
- DescribeTaskSets
- UpdateServicePrimaryTaskSet
- UpdateTaskSet

Marquer les actions

Les API suivantes sont disponibles pour les balises :

- ListTagsForResource
- TagResource
- UntagResource

Limitation des demandes pour l'API Amazon ECS

Amazon Elastic Container Service limite toutes les demandes d'API pour chacune AWS compte par région. Nous procédons ainsi pour garantir des performances constantes et une utilisation équitable du service pour tous les clients Amazon ECS. La limitation garantit que les appels à l'API Amazon ECS ne dépassent pas les quotas de demandes d'API maximum autorisés pour Amazon ECS et les autres AWS services auxquels il s'intègre. Les appels d'API sont soumis aux quotas de demandes, qu'ils proviennent de :

- Une application tierce
- Un outil en ligne de commande
- La console Amazon ECS

Si vous dépassez un quota de limitation de l'API, vous obtenez le `ThrottlingException` code d'erreur.

```
An error occurred (ThrottlingException) when calling the DescribeClusters operation
 (reached max retries: 4): Rate exceeded.
com.amazonaws.services.ecs.model.AmazonECSException: Rate exceeded (Service: AmazonECS;
 Status Code: 400; Error
Code: ThrottlingException; Request ID: 5ed90669-e454-464d-9b2f-6523bc86f537; Proxy:
 null)
```

Comment l'étranglement est-il appliqué

Amazon ECS utilise le algorithme de bucket de jetons pour implémenter la limitation des API. Avec cet algorithme, votre compte dispose d'un compartiment qui contient un nombre spécifique de jetons. Le nombre de jetons dans le compartiment représente votre quota de limitation à chaque seconde.

Amazon ECS examine le taux de soumission de demandes d'API pour toutes les API Amazon ECS de votre compte, par région, et applique deux types de quotas de limitation des API : subi et éclater. Le taux soutenu est le nombre moyen de requêtes d'API autorisées par seconde au fil du temps pour une opération. Le taux de rafale est le nombre maximum de requêtes d'API autorisées par seconde. Avec burst, vous pouvez périodiquement effectuer un nombre de requêtes d'API supérieur au taux soutenu. Amazon ECS limite ensuite les demandes d'API suivantes jusqu'à ce que le taux de demandes d'API autorisées au fil du temps se stabilise pour atteindre le taux soutenu. Dans l'algorithme du bucket de jetons, le capacité maximale du seau signifie la fréquence de rafale et le

taux de remplissage du seau est le taux soutenu. Nous utiliserons ces termes pour vous fournir une illustration de la limitation des requêtes d'API Amazon ECS dans l'exemple suivant.

Le nombre de demandes d'API que vous effectuez est limité et chaque demande supprime un jeton du compartiment de jetons. Par exemple, la taille du seau pour Actions de lecture groupées, comme le `DescribeClusters` API, c'est 50 jetons, vous pouvez donc en créer jusqu'à 50 `DescribeClusters` demandes en une seconde. Si vous dépassez 50 demandes en une seconde, vous êtes limité et les demandes restantes au cours de cette seconde échouent.

Les seaux se rechargent automatiquement à un taux défini. Si le compartiment est inférieur à sa capacité maximale, un nombre défini de jetons y est rajouté toutes les secondes jusqu'à ce qu'il atteigne sa capacité maximale. Si le seau est plein à l'arrivée des jetons de recharge, ils sont jetés. Le compartiment ne peut pas contenir plus que son nombre maximum de jetons. Par exemple, la taille du seau pour Actions de lecture groupées, comme le `DescribeClusters` L'API est de 50 jetons et le taux de recharge est de 20 jetons par seconde. Si vous gagnez 50 `DescribeClusters` Demandes d'API en une seconde, le compartiment est immédiatement réduit à zéro jeton. Le seau est ensuite rempli de 20 jetons par seconde, jusqu'à ce qu'il atteigne sa capacité maximale de 50 jetons. Cela signifie que le seau précédemment vide atteint sa capacité maximale au bout de 2,5 secondes.

Il n'est pas nécessaire d'attendre que le compartiment soit complètement plein pour pouvoir effectuer des requêtes d'API. Vous pouvez utiliser des jetons au fur et à mesure qu'ils sont ajoutés au bucket. Si vous utilisez immédiatement les jetons de recharge, le seau n'atteindra pas sa capacité maximale. Par exemple, la taille du seau pour Actions de lecture groupées, comme le `DescribeClusters` L'API est de 50 jetons et le taux de recharge est de 20 jetons par seconde. Si vous épuisez le compartiment en effectuant 50 demandes d'API par seconde, vous pouvez continuer à effectuer 20 demandes d'API par seconde. Le compartiment ne peut être rempli à sa capacité maximale que si vous effectuez moins de 20 demandes d'API par seconde.

Demandez la taille des seaux de jetons et les tarifs de recharge

À des fins de limitation du taux de demandes, les actions d'API sont regroupées en catégories. Toutes les actions d'API d'une catégorie partagent le même compartiment de jetons. Par exemple, `DescribeClusters` et `ListClusters` Les API partagent Actions de lecture groupées seau, dont la capacité est de 50 et le taux de remplissage est de 20. Cela signifie que le nombre cumulé de demandes d'API pour toutes Actions de lecture groupées est limité par le même quota de fréquence de rafale de 50 requêtes d'API. Ainsi, vous pouvez faire 25 `DescribeClusters` et 25 `ListClusters` Demandes d'API en une seconde, soit 30 `DescribeClusters` et 20

`ListClusters`, soit 50 `DescribeClusters` et 0 `ListClusters`, ou 0 `DescribeClusters` et 50 `ListClusters`, mais tu ne peux pas faire 50 `DescribeClusters` et 50 `ListClusters` demandes en même temps. Le taux soutenu est appliqué de la même manière de manière cumulative à toutes les demandes d'API d'un bucket.

Le tableau suivant indique la capacité du seau (ou rafale) et le taux de remplissage (ou continu) pour tous AWS Régions. Toutes les catégories d'actions d'API appliquent des quotas tarifaires pour chacune AWS compte par région.

Catégorie d'action d'API	Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Actions de modification du cluster	• `CreateCluster` • `DeleteCluster` • `PutClusterCapacityProviders` • `UpdateCluster` • `UpdateClusterSettings`	20	1
Actions de lecture groupées	• `DescribeClusters` • `ListClusters`	50	20
Définition des tâches, modification des actions	• `DeregisterTaskDefinition` • `RegisterDaemonTaskDefinition` • `RegisterTaskDefinition`	20	1
Définition de la tâche : lire les actions	• `DescribeDaemonTaskDefinition` • `DescribeTaskDefinition` • `ListDaemonTaskDefinitions`	50	20

Catégorie d'action d'API	Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
	• `ListTaskDefinitions` • `ListTaskDefinitionFamilies`		
Actions de suppression de définitions de tâches	• `DeleteDaemonTaskDefinition` • `DeleteTaskDefinitions`	5	1
Actions de modification du fournisseur de capacité	• `CreateCapacityProvider` • `DeleteCapacityProvider` • `UpdateCapacityProvider`	10	1
Le fournisseur de capacité lit les actions	• `DescribeCapacityProviders`	50	20
Marquer les actions de modification	• `TagResource` • `UntagResource`	20	10
Marquer les actions de lecture	• `ListTagsForResource`	50	20
Configuration des actions de modification	• `DeleteAccountSetting` • `PutAccountSetting` • `PutAccountSettingDefault`	10	1
Configuration des actions de lecture	• `ListAccountSettings`	50	20

Catégorie d'action d'API	Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Actions de modification des ressources du cluster	• `DeleteAttributes` • `DeregisterContainerInstance` • `ExecuteCommand` • `PutAttributes` • `RunTask`[1] • `StartTask` • `StopTask` • `UpdateContainerAgent` • `UpdateContainerInstancesStates`	100	40
Actions de lecture des ressources du cluster	• `DescribeContainerInstances` • `DescribeTasks` • `ListAttributes` • `ListContainerInstances` • `ListTasks`	100	20
Actions de modification de l'agent	• `RegisterContainerInstance` • `SubmitAttachmentStateChanges` • `SubmitContainerStateChange` • `SubmitTaskStateChange`	200	120

Catégorie d'action d'API	Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Actions de modification du service	• `CreateDaemon` • `CreateExpressGatewayService` • `CreateService` • `DeleteDaemon` • `DeleteExpressGatewayService` • `DeleteService` • `UpdateDaemon` • `UpdateExpressGatewayService` • `UpdateService`	50	5
Actions de lecture du service	• `DescribeDaemon` • `DescribeExpressGatewayService` • `DescribeServices` • `ListDaemons` • `ListServices`	100	20
Actions de déploiement de services	• `DescribeDaemonDeployments` • `DescribeServiceDeployments` • `ListDaemonDeployments` • `ListServiceDeployments`	50	20

Catégorie d'action d'API	Actions	Capacité maximale du compartiment (ou taux en rafale)	Taux de recharge du compartiment (ou taux soutenu)
Actions de révision des services	• `DescribeDaemonRevisions` • `DescribeServiceRevisions`	50	20
Actions de protection des tâches	• `UpdateTaskProtection` • `GetTaskProtection`	200	80
Actions de lecture des ressources du service de cluster	• `ListServicesByNamespace`	10	1

[1] AWS Fargate limite également Amazon ECS `RunTask` API pour les tarifs indiqués ici dans le Guide du développeur Amazon ECS.

Ajustement des quotas de limitation des API

Vous pouvez demander une augmentation des quotas de limitation des API pour votre AWS compte. Pour demander un ajustement de quota, contactezAWS Support Center.

Gestion de la limitation des API

Vous pouvez implémenter une stratégie de nouvelle tentative d'erreur et d'annulation exponentielle pour éviter l'impact des erreurs de limitation sur vos charges de travail. Si vous utilisez AWS SDK, la logique de nouvelle tentative automatique est déjà intégrée et configurable. Vous pouvez consulter les ressources suivantes pour plus de détails :

- Réessais d'erreur et retard exponentiel dans AWS dans le AWS Guide de référence général
- Retard exponentiel et gigue billet de blog
- Délais d'attente, nouvelles tentatives et arrêts avec gigue article dans la bibliothèque Amazon Builder

Les nouvelles tentatives et les annulations aident votre application à se remettre de la limitation, mais elles ne réduisent pas le nombre de requêtes d'API que vous effectuez. Si vous êtes constamment limité, la solution la plus efficace consiste à réduire votre taux de demandes à la source. Prenez en compte les pratiques suivantes avant de demander une augmentation de quota :

- Répartissez le travail prévu. Évitez d'exécuter des tâches par lots, des déploiements ou des boucles d'interrogation pour de nombreuses ressources au même instant (par exemple, au cours de la même minute de cron). Répartissez le travail sur une période plus large et modifiez les heures de début afin que les demandes n'arrivent pas en une seule rafale.
- Interrogez moins fréquemment et réagissez aux événements. Au lieu d'appeler à plusieurs reprises `Describe` et `List` actions pour détecter les changements, allonger votre intervalle d'interrogation ou vous abonner aux changements d'état d'Amazon ECS via Amazon EventBridge et réagir aux changements au fur et à mesure qu'ils se produisent.
- Mettez en cache et réutilisez les résultats. Mettez en cache les réponses des actions de lecture telles que `DescribeTasks` et `ListTasks` et réutilisez-les dans l'ensemble de votre application, plutôt que d'appeler l'API pour chaque tâche ou demande.
- Regroupez vos demandes. Lorsqu'une API accepte plusieurs ressources en un seul appel (par exemple, en transmettant plusieurs ID de tâches à `DescribeTasks`), utilisez-le au lieu de faire une demande par ressource.
- Récupérez les métadonnées des tâches localement. Pour plus d'informations sur une tâche en cours d'exécution, utilisez point final des métadonnées des tâches depuis l'intérieur de la tâche au lieu d'appeler `DescribeTasks`.

Request-rate les quotas protègent les performances du service et une utilisation équitable pour les clients. Si vous avez appliqué ces pratiques et que vous avez toujours besoin d'un taux de demandes plus élevé pour une charge de travail légitime, vous pouvez demander une augmentation comme décrit dans the section called “Ajustement des quotas de limitation des API”.

Garantir l'idempotence

Lorsque vous effectuez une opération de mutation, il se peut qu'une exception apparaisse en raison de délais d'attente ou de problèmes de serveur survenant après la mutation des ressources. Il peut donc être difficile de déterminer si la mutation s'est produite et peut entraîner de multiples tentatives. Toutefois, si l'opération initiale et les nouvelles tentatives suivantes ont effectivement effectué les mutations, vous avez peut-être appliqué des modifications d'empilement ou créé plus de ressources que prévu. L'idempotence garantit qu'une opération ne modifie les ressources qu'une seule fois.

Dans le cas d'une requête idempotente, si la demande d'origine a muté avec succès, toutes les nouvelles tentatives suivantes se terminent avec succès sans effectuer de nouvelle mutation.

Rubriques

- L'idempotence dans Amazon ECS
- Idempotence pour RunTask
- Exemples
- Recommandations de nouvelle tentative pour les requêtes idempotentes

L'idempotence dans Amazon ECS

Les actions d'API suivantes prennent éventuellement en charge l'idempotence à l'aide d'un jeton client. Le correspondant AWS CLI les commandes prennent également en charge l'idempotence à l'aide d'un jeton client. Un jeton client est une chaîne unique qui distingue les majuscules et minuscules. Pour effectuer une demande d'API idempotente à l'aide de l'une de ces actions, spécifiez un jeton client dans la demande. Vous ne devez pas réutiliser le même jeton client pour d'autres demandes d'API. Si vous réessayez une demande qui s'est terminée avec succès en utilisant le même jeton client et les mêmes paramètres, la nouvelle tentative aboutit sans effectuer d'autres actions.

Idempotent à l'aide d'un jeton client

- CreateService

  Le jeton client peut contenir jusqu'à 36 caractères ASCII compris entre 33 et 126 (inclus).
- CreateTaskSet

  Le jeton client peut contenir jusqu'à 36 caractères ASCII compris entre 33 et 126 (inclus).
- RunTask

  Le jeton client peut contenir jusqu'à 64 caractères ASCII compris entre 33 et 126 (inclus).

Types d'idempotence

- cluster — Les requêtes avec le même jeton dans le même cluster sont idempotentes. Par exemple, ClientToken A ne peut être utilisé comme paramètre de requête qu'une seule fois pour `RunTask` requêtes dans le cluster X. `RunTask` les demandes adressées à d'autres clusters sont considérées

comme des demandes distinctes. Par conséquent, vous pouvez utiliser ClientToken A pour a `RunTask` demande pour le cluster Y.

Idempotence pour RunTask

Le `RunTask` L'API prend en charge l'idempotence à l'aide d'un jeton client. Un jeton client est une chaîne unique que vous spécifiez lorsque vous effectuez une demande d'API. Si vous réessayez une demande d'API avec le même jeton client et les mêmes paramètres de requête une fois qu'elle est terminée correctement, le résultat de la demande d'origine est renvoyé. Si vous réessayez une demande réussie en utilisant le même jeton client, mais qu'un ou plusieurs paramètres sont différents, à l'exception de la région ou de la zone de disponibilité, la nouvelle tentative échoue avec un `ConflictException`. Si vous ne spécifiez pas votre propre jeton client, AWS SDK et AWS Command Line Interface génère automatiquement un jeton client pour la demande afin de s'assurer qu'elle est idempotente. Un jeton client peut être n'importe quelle chaîne comprenant jusqu'à 64 caractères ASCII compris entre 33 et 126 (inclus).

Le temps de vivre (TTL) pour le `RunTask` le jeton client est de 24 heures. Vous ne devez pas réutiliser le même jeton client pour différentes demandes. Le TTL maximum du jeton client est valide pour celle des deux valeurs suivantes qui est la plus faible :

- 24 heures
- Durée de vie de la ressource plus une heure

  La durée de vie d'une ressource est l'horodatage auquel la tâche a été créée par rapport à l'horodatage auquel son dernier statut (`lastStatus`) est passé à `STOPPED`. Quand vous utilisez `RunTask` pour lancer plusieurs tâches, la durée de vie de la ressource est égale à la durée de vie de la dernière tâche qui est passée à `STOPPED`.

RunTask règles et réponses relatives aux nouvelles tentatives

Lorsque vous réessayez une demande parce que vous avez reçu une exception 5xx, la réponse réussie de la nouvelle tentative inclut normalement toutes les informations que la demande d'origine aurait renvoyées. Les tâches qui ont été arrêtées pendant moins d'une heure incluent uniquement l'ARN de la tâche, le dernier état et le statut souhaité.

Voici un extrait de la réponse à une nouvelle tentative lorsqu'il y a une tâche en cours d'exécution, une tâche arrêtée et une tâche qui n'a pas pu être lancée.

```
{
  "failures": [
    {
      "arn": "arn:aws:ecs:us-east-1:123456789012:container/4df26bb4-f057-467b-
a079-961675296e64",
      "reason": "RESOURCE:MEMORY"
    }
  ],
  "tasks": [
    {
      "desiredStatus": "RUNNING",
      "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/fdf2c302-468c-4e55-
b884-5331d816e7fb",
      ...
    },
    {
      "taskArn": "arn:aws:ecs:us-east-1:123456789012:task/default/fdf2c302-468c-4e55-
b884-5331d819999",
      "lastStatus": "STOPPED",
      ...
     }
  ]
}
```

Les échecs datant de plus d'une heure incluent uniquement le nombre de tâches ayant échoué.

Exemples

AWS CLI exemples de commandes

Pour faire un AWS CLI commande idempotent, ajoutez `--client-token` option.

Exemple : create-service

Les suivants créer un service La commande utilise l'idempotence car elle inclut un jeton client.

```
aws ecs create-service \
    --cluster MyCluster \
    --service MyService \
    --task-definition MyTaskDefinition:2 \
    --desired-count 2 \
    --launch-type FARGATE \
```

```
    --platform-version LATEST \
    --network-configuration
 "awsvpcConfiguration={subnets=["subnet-12344321"],securityGroups=["sg-12344321"],assignPublicI
 \
    --client-token 550e8400-e29b-41d4-a716-44665544
```

Exemple : create-task-set

Les suivants créer un ensemble de tâches La commande utilise l'idempotence car elle inclut un jeton client.

```
aws ecs create-task-set \
    --cluster MyCluster \
    --service MyService \
    --task-definition MyTaskDefinition:2 \
    --network-configuration
 "awsvpcConfiguration={subnets=["subnet-12344321"],securityGroups=["sg-12344321"]}" \
    --client-token 550e8400-e29b-41d4-a716-44665544
```

Exemple : run-task

Les suivants exécuter une tâche La commande utilise l'idempotence car elle inclut un jeton client.

```
aws ecs run-task \
    --cluster MyCluster \
    --task-definition MyTaskDefinition:2 \
    --client-token 550e8400-e29b-41d4-a716-446655440000
```

Exemples de demandes d'API

Pour rendre une requête d'API idempotente, ajoutez `clientToken` paramètre.

Exemple : CreateService

Les suivants CreateService La requête d'API utilise l'idempotence car elle inclut un jeton client.

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 87
```

```
X-Amz-Target: AmazonEC2ContainerServiceV20141113.CreateService
X-Amz-Date: 20150429T170125Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "serviceName": "MyService",
  "taskDefinition": "MyTaskDefinition:2",
  "desiredCount": 10,
   "capacityProviderStrategy": [
      {
         "base": "number",
         "capacityProvider": "FARGATE",
         "weight": 1
      }
   ],
   "capacityProviderStrategy": [
      {
         "base": "number",
         "capacityProvider": "FARGATE_SPOT",
         "weight": 1
      }
   ],
   "clientToken": "550e8400-e29b-41d4-a716-44665544"
}
```

Exemple : CreateTaskSet

Les suivants CreateTaskSet La requête d'API utilise l'idempotence car elle inclut un jeton client.

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 87
X-Amz-Target: AmazonEC2ContainerServiceV20141113.CreateTaskSet
X-Amz-Date: 20150429T170125Z
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "serviceName": "MyService",
  "taskDefinition": "mytask:1",
  "desiredCount": 1,
   "capacityProviderStrategy": [
```

```
      {
         "base": "number",
         "capacityProvider": "FARGATE",
         "weight": 1
      }
   ],
   "capacityProviderStrategy": [
      {
         "base": "number",
         "capacityProvider": "FARGATE_SPOT",
         "weight": 1
      }
   ],
    "clientToken": "550e8400-e29b-41d4-a716-44665544"
}
```

Exemple : RunTask

Les suivants RunTask La requête d'API utilise l'idempotence car elle inclut un jeton client.

```
POST / HTTP/1.1
Host: ecs.us-east-1.amazonaws.com
Accept-Encoding: identity
Content-Length: 45
X-Amz-Target: AmazonEC2ContainerServiceV20141113.RunTask
X-Amz-Date: 20161121T215740Z
User-Agent: aws-cli/1.11.13 Python/2.7.12 Darwin/16.1.0 botocore/1.4.66
Content-Type: application/x-amz-json-1.1
Authorization: AUTHPARAMS

{
  "count": 1,
  "taskDefinition": "mytask:1",
  "clientToken": "550e8400-e29b-41d4-a716-446655440000"
}
```

Recommandations de nouvelle tentative pour les requêtes idempotentes

La table suivante présente certaines réponses courantes que vous pouvez obtenir pour les demandes d’API idempotentes et fournit des recommandations pour réessayer.

Réponse	Recommandation	Commentaires
200 (OK)	Ne pas réessayer	La demande d'origine s'est terminée avec succès. Toutes les tentatives suivantes sont renvoyées avec succès.
Codes de réponse de la série 400 (erreurs du client)	Ne pas réessayer	La demande présente un problème, parmi les suivants : • Elle inclut un paramètre ou une combinaison de paramètres invalide. • Elle utilise une action ou une ressource pour laquelle vous n'avez pas les autorisations nécessaires. • Elle utilise une ressource qui est en train de changer d'état. Si la demande concerne une ressource en train de changer d'état, il est possible que la nouvelle tentative aboutisse.
Codes de réponse de la série 500 (erreurs de serveur)	Réessayer	L'erreur est causée par un AWS problème côté serveur et est généralement transitoire. Répétez la demande avec une stratégie de sauvegarde appropriée.

Historique du document

Le tableau suivant décrit les principales mises à jour et les nouvelles fonctions pour le Guide du développeur Amazon Elastic Container Service. Nous mettons aussi la documentation à jour régulièrement pour prendre en compte les commentaires qui nous sont envoyés.

Modifier	Description	Date
Ajoutez des autorisations à Amazon ECS_FullAccess politique gérée.	Le `AmazonECS_FullAccess` la politique IAM gérée a été mise à jour pour inclure le `cloudformation:TagResource` et `cloudformation:UntagResource` autorisations. Ces autorisations permettent d'ajouter des balises à CloudFormation piles lorsqu'une pile est créée ou mise à jour.	2 septembre 2026
Ajout d'autorisations à la politique gérée AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour pour inclure l'autorisation `autoscaling:TerminateInstanceInAutoScalingGroup`. Cette autorisation permet à Amazon ECS de mettre fin aux instances Amazon EC2 dans les ASG clients qui ne parviennent pas à s'enregistrer auprès du cluster Amazon ECS.	24 avril 2026
Mettre à jour les autorisations pour `AmazonECSInfrastructureRolePolicyForManagedInstances`.	Les autorisations ont été mises à jour pour maintenir la cohérence du nom du service EC2 entre les partitions. Cette mise à jour ne modifie pas le modèle d'autorisation existant et n'accorde pas d'accès supplémentaire.	26 février 2026
Mettre à jour les autorisations pour `AmazonECSInfrastructureRolePolicyForManagedInstances`.	Le `AmazonECSInfrastructureRolePolicyForManagedInstances` la politique a été mise à jour pour prendre en charge les réservations de capacité sur les instances gérées Amazon ECS en ajoutant les autorisations suivantes : •	24 février 2026

Modifier	Description	Date
	`resource-groups:ListGroupResources` l'autorisation a été ajoutée pour permettre à Amazon ECS de récupérer un groupe de réservations de capacité. Pour exécuter cette commande, les autorisations suivantes sont également requises : `cloudformation:DescribeStacks` , `cloudformation:ListStackResources` , `tag:GetResources` • `ec2:DescribeCapacityReservations` l'autorisation a été ajoutée pour permettre à Amazon ECS de recueillir des informations sur les réservations de capacité nécessaires à l'intégration aux instances gérées Amazon ECS. • `ec2:RunInstances` l'autorisation a été modifiée pour inclure Amazon ResourceGroups ressources de groupe permettant à Amazon ECS de fournir des réservations de capacité Amazon EC2 ciblées.	
Ajout d'autorisations à la politique gérée AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour pour inclure l'autorisation `ssmmessages:OpenDataChannel` . Cette autorisation permet à Amazon ECS d'ouvrir des canaux de données pour les sessions ECS Exec.	20 janvier 2026

Modifier	Description	Date
Mettre à jour les autorisations pour `AmazonECSInfrastructureRolePolicyForManagedInstances` .	Le `AmazonECSInfrastructureRolePolicyForManagedInstances` la politique a été mise à jour pour modifier la `CreateFleet` autorisations. Les conditions basées sur les ressources pour les sous-réseaux, les groupes de sécurité et les images EC2 ont été supprimées pour les raisons suivantes : • Les sous-réseaux et les groupes de sécurité ne sont ni créés ni gérés par Amazon ECS, ils ne contiennent donc pas les informations requises `AmazonECSManaged` balises. • Bien qu'Amazon ECS possède et gère les images EC2, les balises ne sont pas visibles pour les clients, ce qui empêche l'application de politiques basées sur les ressources. Cette modification garantit que la politique fonctionne correctement avec les ressources qui ne possèdent pas les balises de gestion ECS attendues.	15 décembre 2025
Correct instances gérées politique à vie	Mise à jour de la documentation sur les instances gérées afin de clarifier la terminologie relative à la durée de vie des instances et de prolonger la durée de vie maximale des instances de 14 à 21 jours.	11 décembre 2025
Ajout d'autorisations à la politique gérée AmazonECSServiceRolePolicy.	La politique IAM gérée `AmazonECSServiceRolePolicy` a été mise à jour pour inclure l'autorisation `ec2:DescribeInstanceEventWindows` . Cette autorisation permet à Amazon ECS de récupérer les fenêtres d'événements Amazon EC2 pour les services et les clusters associés à Event Windows.	20 novembre 2025

Modifier	Description	Date
Ajout d'autorisations à `AmazonECSServiceRolePolicy` .	Le `AmazonECSServiceRolePolicy` a été mis à jour pour inclure le `ec2:AssociateTrunkInterface` et `ec2:DisassociateTrunkInterface` autorisations pour activer Amazon ECSto pour provisionner et déprovisionner l'ENI de tâche.	14 novembre 2025
Mettre à jour the section called "`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`" politique	Mis à jour the section called "`AmazonECSInfrastructureRolePolicyForServiceConnectTransportLayerSecurity`" politique permettant à Amazon ECS de surveiller le cycle de vie des secrets créés par Amazon ECS.	13 novembre 2025
Déploiements canary d'Amazon ECS	Les déploiements Amazon ECS Canary vous permettent de tester de nouvelles versions d'applications en acheminant un faible pourcentage du trafic vers la nouvelle version tout en conservant la majorité sur la version stable. Pour plus d'informations, consultez Déploiements canary d'Amazon ECS.	28 octobre 2025
Déploiements linéaires Amazon ECS	Les déploiements linéaires Amazon ECS vous permettent de transférer progressivement le trafic de la révision actuelle du service vers une nouvelle révision par incréments de pourcentage égaux, ce qui permet un déploiement contrôlé avec une surveillance à chaque étape. Pour plus d'informations, consultez Déploiements linéaires Amazon ECS.	28 octobre 2025

Modifier	Description	Date
Régions des instances gérées Amazon ECS	Les instances gérées Amazon ECS sont désormais disponibles dans d'autres régions. Pour plus d'informations, voir Architecte pour les instances gérées Amazon ECS.	27 octobre 2025
AWS pour Fluent Bit version 3.0.0	AWS pour la version Fluent Bit `3.0.0` est maintenant disponible, basé sur la version Fluent Bit `4.1.1` et AL2023. Pour plus d'informations, consultez AWS for Fluent Bit référentiels d'images pour Amazon ECS.	14 octobre 2025
FireLens support aux utilisateurs non root du conteneur	Amazon ECS prend en charge l'exécution du FireLens conteneur en tant qu'utilisateur non root. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	13 octobre 2025
Capture d'événements Amazon ECS dans la console	La console Amazon ECS fournit une fonctionnalité de capture d'événements qui stocke Amazon ECS-generated événements, tels que les actions de service et les changements d'état des tâches, pour CloudWatch Se connecte EventBridge. Pour plus d'informations, voir Événements du cycle de vie des tâches.	2 octobre 2025
Amazon ECS prend en charge les instances gérées Amazon ECS pour exécuter des charges de travail conteneurisées	Amazon ECS prend en charge les instances gérées Amazon ECS pour exécuter des charges de travail conteneurisées. Pour plus d'informations, consultez la section Instances gérées Amazon ECS.	30 septembre 2025
Ajout de la nouvelle politique the section called "`AmazonECSInstanceRolePolicyForManagedInstances`"	Nouveau ajouté AmazonECSInstanceRolePolicyForManagedInstances politique qui autorise les instances gérées par Amazon ECS à s'enregistrer auprès des clusters Amazon ECS.	30 septembre 2025

Modifier	Description	Date
Ajout de la nouvelle politique the section called “AmazonECSInfrastructureRolePolicyForManagedInstances”	Nouveau ajouté AmazonECSInfrastructureRolePolicyForManagedInstances politique qui permet à Amazon ECS d'accéder à la création et à la gestion des ressources gérées par Amazon EC2.	30 septembre 2025
Prise en charge de l'exécution des charges de travail Amazon ECS dans un IPv6-only configuration	Les charges de travail Amazon ECS peuvent désormais être exécutées dans un IPv6-only configuration qui permet aux tâches de communiquer exclusivement via IPv6. Pour plus d’informations, consultez les sections Options de mise en réseau des tâches Amazon ECS pour le type de lancement EC2 et Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate.	29 septembre 2025
Prise en charge de l’utilisation des espaces de noms AWS Cloud Map partagés pour Amazon ECS Service Connect	Amazon ECS Service Connect prend désormais en charge l’utilisation d’espaces de noms AWS Cloud Map partagés pour la communication de service à service intercomptes. Pour plus d’informations, consultez la section Amazon ECS Service Connect avec espaces de noms AWS Cloud Map partagés.	12 septembre 2025
Amazon ECS améliore la modification des définitions de tâches dans le AWS console avec Amazon Q	Pour plus d’informations, consultez la section Utilisation d’Amazon Q pour fournir des recommandations de définition de tâches dans la console Amazon ECS	10 septembre 2025
Support pour l’exécution des commandes ECS Exec depuis la console	Vous pouvez désormais exécuter les commandes ECS Exec directement depuis la console Amazon ECS. Pour plus d’informations, consultez la section Exécution des commandes ECS Exec à partir de la console.	4 septembre 2025

Modifier	Description	Date
Ajout de la nouvelle politique the section called “AmazonECSInfrastructureRolePolicyForManagedInstances”	Nouveau ajouté AmazonECSInfrastructureRolePolicyForManagedInstances politique qui permet à Amazon ECS d'accéder à la création et à la gestion des ressources gérées par Amazon EC2.	31 août 2025
Ajout d’autorisations à la politique gérée AmazonEC2ContainerServiceforEC2Role.	La politique IAM gérée `AmazonEC2ContainerServiceforEC2Role` a été mise à jour pour inclure l’autorisation `ecs:ListTagsForResource`. Cette autorisation permet à l’agent Amazon ECS de récupérer les balises des tâches et des instances de conteneurs via le point de terminaison des métadonnées de tâche (`${ECS_CONTAINER_METADATA_URI_V4}/taskWithTags`). Pour plus d’informations, consultez AmazonEC2ContainerServiceforEC2Role.	4 août 2025
Ajout d’autorisations à AmazonECSInfrastructureRolePolicyForLoadBalancers.	La politique IAM gérée `AmazonECSInfrastructureRolePolicyForLoadBalancers` a été mise à jour avec de nouvelles autorisations pour la description, l’enregistrement et l’annulation de l’enregistrement des groupes cibles.	25 juillet 2025
Prise en charge de la mise à jour d’un contrôleur de déploiement de service	Amazon ECS prend en charge la mise à jour d’un contrôleur de déploiement de service, ce qui vous permet de passer d’un type de déploiement à un autre pour un service existant. Pour plus d’informations, consultez la section Mise à jour des paramètres du service Amazon ECS.	15 juillet 2025

Modifier	Description	Date
Amazon ECS natif blue/green déploiements	Amazon ECS prend en charge les versions natives blue/green déploiements, ce qui vous permet de valider les nouvelles révisions de service avant de diriger le trafic de production vers ces derniers. Pour plus d'informations, voir Amazon ECS blue/green déploiements.	15 juillet 2025
Ajout de la nouvelle politique gérée AmazonECSInfrastructureRolePolicyForLoadBalancers.	Cette politique donne accès à d'autres AWS les ressources de service nécessaires pour gérer les équilibreurs de charge associés aux charges de travail Amazon ECS en votre nom. Pour plus d'informations, consultez AmazonECSInfrastructureRolePolicyForLoadBalancers.	15 juillet 2025
Prise en charge du manifeste v2 d'index SOCI	Les clients Fargate souhaitant utiliser SOCI ne pourront utiliser que le manifeste v2 d'index SOCI. Les clients existants qui ont déjà utilisé SOCI sur Fargate peuvent continuer à utiliser le manifeste v1 d'index SOCI mais nous leur conseillons vivement de passer à le manifeste v2 d'index SOCI. Pour plus d'informations, consultez la section Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI).	2 juillet 2025
Ajout de l'ID de tâche dans les événements d'action de service générés en raison de dysfonctionnements.	Pour plus d'informations, consultez la section Messages d'événement relatifs au dysfonctionnement d'un service Amazon ECS.	30 juin 2025

Modifier	Description	Date
Mise à jour vers le mode par défaut du pilote de journalisation.	Le mode de journalisation par défaut pour Amazon ECS a été mis à jour de `blocking` à `non-blocking` . Vous pouvez remplacer ce mode par défaut en spécifiant un `mode` dans votre définition de conteneur `logConfiguration` ou configurer votre propre mode par défaut en définissant le paramètre de compte `defaultLogDriverMode` à l'aide de la console Amazon ECS ou de l' AWS CLI. Pour plus d'informations, consultez la section Mode par défaut du pilote de journalisation.	25 juin 2025
Ajout d'autorisations à AmazonECSServiceRolePolicy.	Le `AmazonECSServiceRolePolicy` la politique IAM gérée a été mise à jour avec de nouvelles AWS Cloud Map autorisations qu'Amazon ECS peut mettre à jour AWS Cloud Map attributs de service pour les services gérés par Amazon ECS.	24 juin 2025
Assistance pour Amazon ECS-optimized AMI complète pour Windows Server 2025 et Amazon ECS-optimized AMI principale Windows Server 2025.	Pour plus d'informations, voir Amazon ECS-optimized AMI Windows.	20 juin 2025
Prise en charge de la mise à jour d'une stratégie de fournisseur de capacité de service.	Vous pouvez désormais mettre à jour une stratégie de fournisseur de capacité de service. Pour plus d'informations, consultez la section Paramètres de mise à jour du service.	12 juin 2025

Modifier	Description	Date
Ajouter des autorisations à AmazonECSInfrastructureRolePolicyForVolumes	La politique `AmazonECSInfrastructureRolePolicyForVolumes` a été mise à jour pour ajouter l'autorisation `ec2:DescribeInstances`. Cette autorisation permet d'éviter les conflits de noms d'appareils pour les volumes Amazon EBS attachés à des tâches Amazon ECS exécutées sur la même instance de conteneur.	2 juin 2025
Prise en charge du taux d'initialisation des volumes Amazon EBS associés aux tâches Amazon ECS.	Vous pouvez désormais définir un `volumeInitializationRate` pour déterminer le taux, en MiB/s, au cours de laquelle les données sont extraites d'un instantané d'un volume Amazon EBS existant afin de créer de nouveaux volumes à joindre aux tâches Amazon ECS. Pour plus d'informations, voir volumeInitializationRate.	13 mai 2025
Prise en charge de l'arrêt du déploiement d'un service.	Vous pouvez désormais arrêter le déploiement d'un service. Pour plus d'informations, consultez la section Arrêt des déploiements de service Amazon ECS.	2 mai 2025
Prise en charge des points de terminaison à double pile Amazon ECS.	Vous pouvez désormais utiliser des points de terminaison à double pile pour interagir avec Amazon ECS. Dual-stack les terminaux prennent en charge les demandes adressées à Amazon ECS via le protocole Internet version 4 (IPv4) et le protocole Internet version 6 (IPv6). Pour plus d'informations, consultez la section Utilisation des points de terminaison à double pile Amazon ECS.	30 avril 2025

Modifier	Description	Date
Prise en charge de la configuration d'un mode de pilote de journal par défaut au niveau du compte dans une Région AWS.	Vous pouvez désormais configurer un mode de livraison par défaut des messages de journalisation depuis un conteneur vers le pilote de journalisation que vous spécifiez dans votre `logConfiguration` à l'aide du paramètre `defaultLogDriverMode` du compte. Pour plus d'informations, consultez la section Mode par défaut du pilote de journalisation.	17 avril 2025
Prise en charge de la migration d'un service utilisant un format ARN court vers un format ARN long.	Vous pouvez désormais procéder à la migration d'un service au format ARN court vers le paquet au format ARN long. Pour plus d'informations, consultez la section Procéder à la migration d'un ARN court Amazon ECS vers un ARN long.	13 février 2025
Prise en charge de trois types d'injection de pannes sur Amazon ECS et Fargate.	Vous pouvez désormais utiliser l'injection de pannes au niveau du port réseau blackhole, de la latence réseau et de la perte de paquets réseau vers Amazon ECS et Fargate. Pour plus d'informations, consultez la section Utilisation de AWS Fault Injection Service pour tester vos charges de travail Amazon ECS.	19 décembre 2024
Assistance pour CloudWatch Container Insights avec une observabilité améliorée.	Vous pouvez utiliser CloudWatch Container Insights avec une observabilité améliorée. Pour plus d'informations, consultez la section Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.	2 décembre 2024
Prise en charge de la mise à l'échelle prédictive.	Vous pouvez utiliser la mise à l'échelle prédictive pour anticiper les besoins futurs et augmenter de manière proactive les tâches de votre service en fonction des besoins. Pour plus d'informations, voir Utiliser des modèles historiques pour redimensionner CloudWatch services avec mise à l'échelle prédictive.	21 novembre 2024

Modifier	Description	Date
Prise en charge du rééquilibrage des services.	Amazon ECS peut désormais équilibrer automatiquement vos tâches de service entre les zones de disponibilité. Pour plus d'informations, consultez Équilibrage d'un service Amazon ECS entre les zones de disponibilité.	19 novembre 2024
Prise en charge de la configuration de résolution des balises d'image de conteneur en condensés d'image.	Vous pouvez désormais configurer la résolution des balises d'image de conteneur en condensés pour chaque conteneur dans les tâches qui font partie d'un service. Pour plus d'informations, consultez Résolution des images de conteneurs et `versionConsistency`.	19 novembre 2024
Ajout de la nouvelle politique IAM `AmazonECSInfrastructureRolePolicyForVpcLattice`.	Le `AmazonECSInfrastructureRolePolicyForVpcLattice` la politique donne accès à d'autres AWS ressources de service requises pour gérer la fonctionnalité VPC Lattice dans les charges de travail Amazon ECS en votre nom.	18 novembre 2024
VPC Lattice est désormais disponible pour les services Amazon ECS.	Vous pouvez désormais utiliser VPC Lattice pour standardiser votre connectivité, votre sécurité et votre observabilité de service à service. Pour plus d'informations, consultez la section Utilisation d'Amazon VPC Lattice pour connecter, observer et sécuriser vos services Amazon ECS.	18 novembre 2024
Prise en charge de l'historique des services à l'aide des déploiements et des révisions de service.	Vous pouvez utiliser les déploiements de service et les révisions de service pour consulter l'historique des déploiements. Pour plus d'informations, consultez les sections Affichage de l'historique des services à l'aide des déploiements de service Amazon ECS et Révisions de service Amazon ECS.	30 octobre 2024

Modifier	Description	Date
Prise en charge des volumes Amazon EBS pour le type de lancement EC2 utilisant le système d'exploitation Windows.	Amazon ECS prend en charge les volumes Amazon EBS associés à des tâches exécutées sur des instances EC2 avec le système d'exploitation Windows. Pour plus d'informations, consultez la section Utilisation des volumes Amazon EBS avec Amazon ECS.	24 octobre 2024
Ajoutez des autorisations à AmazonECSInfrastructureRolePolicyForVolumes.	La politique `AmazonECSInfrastructureRolePolicyForVolumes` a été mise à jour pour permettre aux clients de créer un volume Amazon EBS à partir d'un instantané. Pour plus d'informations, consultez AmazonECSInfrastructureRolePolicyForVolumes.	10 octobre 2024
Fargate Spot est disponible pour Linux ARM	La prise en charge de Fargate Spot sur Linux ARM a été ajoutée à Amazon ECS.	6 septembre 2024
Politique de redémarrage des conteneurs individuels dans les tâches Amazon ECS.	Vous pouvez activer des stratégies de redémarrage pour les conteneurs essentiels et non essentiels spécifiés dans votre définition de tâche afin de surmonter plus rapidement les défaillances transitoires et de maintenir la disponibilité des tâches. Pour plus d'informations, consultez Redémarrage de conteneurs individuels dans les tâches Amazon ECS à l'aide de politiques de redémarrage de conteneurs.	15 août 2024
Autorisations ajoutées à the section called "Amazon ECS_FullAccess"	La politique `AmazonECS_FullAccess` a été mise à jour pour ajouter des autorisations `iam:PassRole` pour les rôles IAM pour un rôle nommé `ecsInfrastructureRole`. Il s'agit du rôle IAM par défaut créé par Console de gestion AWS qui est destiné à être utilisé comme rôle d'infrastructure ECS permettant à Amazon ECS de gérer les volumes Amazon EBS associés aux tâches ECS.	13 août 2024

Modifier	Description	Date
Ajout de la prise en charge d'Amazon Linux 2023, de CentOS Stream 9, de Fedora 40 et d'Ubuntu 24 pour ECS Anywhere	Le support pour les systèmes d'exploitation Amazon Linux 2023, CentOS Stream 9, Fedora 40 et Ubuntu 24 a été ajouté à ECS Anywhere. Pour plus d'informations, consultez Systèmes d'exploitation et architectures système pris en charge.	23 juillet 2024
Résolution d'image de conteneur pour les services utilisant le type de déploiement de mise à jour propagée (ECS).	Pour garantir que toutes les tâches d'un service qui utilise le contrôleur de déploiement de mise à jour propagée utilisent la même image de conteneur, Amazon ECS résout les noms d'images de conteneur et toutes les balises d'image spécifiées dans la définition de tâche en condensés d'images de conteneur. Pour plus d'informations, consultez la section Résolution d'image de conteneur.	10 juillet 2024
Prise en charge de Debian 11 et Debian 12 ajoutée pour ECS Anywhere	La prise en charge des systèmes d'exploitation Debian 11 et Debian 12 a été ajoutée à ECS Anywhere. Pour plus d'informations, consultez Systèmes d'exploitation et architectures système pris en charge.	28 mars 2024
Prise en charge de gMSA pour les conteneurs Linux sur Fargate	Amazon ECS prend en charge l'authentification Active Directory pour les conteneurs Linux sur Fargate par l'intermédiaire d'un type de compte de service spécial appelé gSMA (group Managed Service Account). Pour plus d'informations, consultez la section Utilisation de gMSA pour les conteneurs Linux sur Fargate.	5 mars 2024
CloudWatch mesures ajoutées pour les volumes Amazon EBS associés à des tâches	Amazon ECS publie désormais CloudWatch des mesures relatives à l'utilisation du stockage Amazon EBS pour les tâches auxquelles un volume Amazon EBS est attaché. Pour plus d'informations, voir Amazon ECS CloudWatch indicateurs.	8 février 2024

Modifier	Description	Date
Service Connect TLS	Vous pouvez désormais utiliser TLS avec Service Connect.	22 janvier 2024
Politique gérée Service Connect TLS	Nouveau ajouté AmazonECSInfrastructureRole PolicyForServiceConnectTransportLayerSecurity politique.	22 janvier 2024
Mise à jour de la configuration du délai d'expiration de Service Connect	La configuration du délai d'expiration de Service Connect peut désormais être mise à jour et inclut deux paramètres facultatifs : `idleTimeout` et `perRequestTimeout` .	22 janvier 2024
Drainage d'instance gérée Amazon ECS	Vous pouvez utiliser le drainage d'instance gérée Amazon ECS pour faciliter la résiliation progressive des instances Amazon ECS.	19 janvier 2024
Ajout de la prise en charge d'Ubuntu 22 pour ECS Anywhere	Le support pour le système d'exploitation Ubuntu 22 a été ajouté à ECS Anywhere. Pour plus d'informations, consultez Systèmes d'exploitation et architectures système pris en charge.	16 janvier 2024
Ajout de la politique IAM `AmazonECSInfrastructureRole PolicyForVolumes`	Le `AmazonECSInfrastructureolePolicyForVolumes` a été ajouté. La politique accorde les autorisations nécessaires à Amazon ECS pour AWS Appels d'API pour gérer les volumes Amazon EBS associés aux charges de travail Amazon ECS.	11 janvier 2024

Modifier	Description	Date
Volume de données Amazon EBS pour la tâche Amazon ECS	Vous pouvez configurer un volume de données Amazon EBS par tâche à rattacher à des tâches Amazon ECS autonomes ou à des tâches gérées par un service ECS pendant le déploiement. La configuration d'un volume lors du déploiement vous permet de créer des définitions de tâches réutilisables qui ne sont pas limitées à des types de volumes ou à des paramètres spécifiques. Les volumes Amazon EBS fournissent un stockage par blocs hautement disponible, rentable, durable et performant pour les charges de travail conteneurisées gourmandes en données.	11 janvier 2024
La console Amazon ECS classique a atteint sa fin de vie	La console Amazon ECS a atteint la fin de son cycle de vie.	4 décembre 2023
Politique mise à jour	La politique IAM gérée AmazonECSServiceRolePolicy a été mise à jour avec de nouvelles autorisations `events` et des autorisations supplémentaires `autoscaling` et `autoscaling-plans` .	4 décembre 2023
Prise en charge de la surveillance d'exécution	Vous pouvez utiliser la surveillance d'exécution pour surveiller vos charges de travail Amazon ECS afin d'identifier les comportements malveillants ou non autorisés. Pour plus d'informations, consulter la section Surveillance d'exécution.	26 novembre 2023
Stratégie mise à jour	Le `AmazonECSServiceRolePolicy` la politique IAM gérée a été mise à jour pour permettre l'accès au AWS Cloud Map `DiscoverInstancesRevision` API.	4 octobre 2023
AWS Fargate configuration de retrait des tâches	Vous pouvez configurer la période d'attente avant le retrait des tâches Fargate. Pour plus d'informations, veuillez consulter Maintenance des tâches AWS Fargate (langue française non garantie).	5 septembre 2023

Modifier	Description	Date
Paramètres de définitio n de tâches supplémen taires dans AWS Fargate	AWS Fargate ajoute la prise en charge de `pidMode` et `systemControls` dans la version de la plateform e Linux `1.4.0`. Pour plus d'informations, veuillez consulter Définitions de tâches (langue française non garantie).	9 août 2023
Refonte de la page de définition des tâches de la console Amazon ECS	La page de définition des tâches de la console Amazon ECS a été repensée et contient désormais des options supplémentaires. Pour plus d'informations, veuillez consulter Création d'une définition de tâche à l'aide de la console (langue française non garantie).	26 juillet 2023
Fargate prend en charge le chargement différé avec les index Seekable OCI.	AWS Fargate présente les indices Seekable OCI (SOCI). Avec SOCI, les conteneurs ne passent que quelques secondes à extraire l'image avant de pouvoir démarrer, ce qui laisse le temps de configurer l'environ nement et d'instancier l'application pendant que l'image est téléchargée en arrière-plan. Pour plus d'informations, consultez la section Chargement différé d'images de conteneurs à l'aide de Seekable OCI (SOCI).	17 juillet 2023
Prise en charge améliorée pour gMSA sous Linux et Windows	La définition de tâche comporte un nouveau champ `credentialSpecs` pour gMSA sous Linux et Windows. Un nouveau didacticiel complet pour les gMSA sans domaine sous Windows a été ajouté, veuillez consulter Didacticiel : utilisation de conteneur s Windows avec gMSA sans domaine à l'aide de l' AWS CLI (langue française non garantie). Pour plus d'informations, veuillez consulter Utilisation de gMSAs pour les conteneurs Linux et Utilisation de gMSA pour les conteneurs Windows.	14 juillet 2023

Modifier	Description	Date
Documentation améliorée sur les versions de l'agent ECS	La documentation relative aux versions de l'agent Amazon ECS a été mise à jour. Nous vous recommandons d'utiliser la version `v20.10.13` de Docker, ou une version plus récente, avec la dernière version de l'agent de conteneur Amazon ECS. Les versions publiées et les modifications apportées à l'agent sont disponibles sur GitHub. Pour plus d'informations, veuillez consulter Versions de l'agent de conteneur Linux Amazon ECS (langue française non garantie).	20 juin 2023
Disponibilité régionale mise à jour pour la prise en charge d'ARM64 par Fargate	La disponibilité régionale pour la prise en charge d'ARM64 par Fargate a été mise à jour. Pour plus d'informations, consultez Considérations.	19 juin 2023
Amélioration de la documentation relative à l'autoscaling de cluster	La documentation relative à la mise à l'échelle Amazon ECS d'Amazon EC2 Auto Scaling présente des améliorations significatives en termes de précision et de lisibilité. Pour plus d'informations, veuillez consulter Autoscaling de cluster Amazon ECS (langue française non garantie).	4 mai 2023
Autorisation de balisage pour la création de ressources.	Les utilisateurs doivent être autorisés à effectuer les actions qui créent la ressource, comme `ecsCreateCluster` . Lorsque vous créez une ressource et que vous spécifiez des balises pour cette ressource, AWS effectue une autorisation supplémentaire pour vérifier qu'il existe des autorisations pour créer des balises. Pour plus d'informations, veuillez consulter Autorisation de balisage et Octroi de l'autorisation de baliser les ressources lors de la création.	18 avril 2023

Modifier	Description	Date
Prise en charge de gMSA pour des conteneurs Linux sur EC2	Vous pouvez utiliser gMSA pour vous authentifier auprès d'Active Directory pour des conteneurs Linux sur EC2. Pour plus d'informations, veuillez consulter Utilisation de gMSAs pour les conteneurs Linux.	14 avril 2023
Prise en charge du stockage éphémère pour les conteneurs Windows sur AWS Fargate	Vous pouvez utiliser le stockage éphémère pour les conteneurs Windows sur AWS Fargate. Pour plus d'informations, veuillez consulter Stockage des tâches Fargate (langue française non garantie).	14 avril 2023
AWS Cost Management prise en charge des données CUR au niveau des tâches	Vous pouvez activer l'utilisation des ressources et les coûts au niveau des tâches dans les rapports d'utilisation et de coût. Cela ajoute des données de répartition des coûts fractionnés pour les tâches exécutées sur AWS Fargate et EC2. Pour plus d'informations, consultez Task-level Rapports sur les coûts et l'utilisation.	12 avril 2023
Amazon Linux 2023 ECS-optimized AMI	Vous pouvez déployer des charges de travail sur Amazon Linux 2023 (Amazon) ECS-optimized AMI. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	10 avril 2023
AWS Fargate Norme fédérale de traitement de l'information (FIPS) 140	Vous pouvez déployer des charges de travail sur Amazon ECS sur AWS Fargate d'une manière conforme à la norme fédérale de traitement de l'information (FIPS) 140. Pour plus d'informations, consultez AWS Fargate Norme fédérale de traitement de l'information (FIPS-140).	10 avril 2023
Suppression d'une définition de tâche	Vous pouvez supprimer une définition de tâche à l'aide de la console Amazon ECS, du SDK et de la AWS CLI. Pour plus d'informations, veuillez consulter Suppression de l'enregistrement d'une révision de définition de tâche à l'aide de la console et Définitions de tâches.	24 février 2023

Modifier	Description	Date
AWS Fargate recommandations de service dans Compute Optimizer	AWS Compute Optimizer génère des recommandations relatives à la taille des tâches et des conteneurs en fonction de l'utilisation des tâches en cours d'exécution dans les services Amazon ECS sur AWS Fargate. Pour plus d'informations, veuillez consulter Affichage des recommandations pour les services Amazon ECS sur Fargate (langue française non garantie).	27 janvier 2023
Console Amazon ECS	La nouvelle console Amazon ECS est désormais la console par défaut. Pour plus d'informations, consultez Nouvelle console Amazon ECS.	19 janvier 2023
Politique IAM `AmazonECS_FullAccess` mise à jour	La politique IAM `AmazonECS_FullAccess` est mise à jour pour inclure des autorisations permettant d'ajouter des balises aux équilibreurs de charge lors de la création. Pour plus d'informations, consultez Amazon ECS_FullAccess.	4 janvier 2023
Utiliser CloudWatch alarmes pour détecter les échecs de déploiement du service Amazon ECS	Vous pouvez configurer Amazon ECS pour définir l'échec du déploiement lorsqu'il détecte qu'un CloudWatch l'alarme est passée à l'état ALARME. Pour plus d'informations, consultez the section called “Déploiements de mise à jour propagée”.	19 décembre 2022
Prise en charge du mappage de ports des conteneurs	Vous pouvez définir une plage de numéros de port du conteneur liée à la plage de ports hôtes mappés dynamiquement. Pour plus d'informations, consultez the section called “Cartographie des ports”.	15 décembre 2022
Disponibilité générale d'Amazon ECS Service Connect	Cette fonctionnalité ajoute la découverte de services et un maillage de services contrôlés par les déploiements du service Amazon ECS. Pour plus d'informations, consultez the section called “ Connexion au service”.	27 novembre 2022

Modifier	Description	Date
La nouvelle expérience de la console Amazon ECS pour les définitions de tâches est mise à jour	La nouvelle expérience de la console Amazon ECS contient désormais un éditeur JSON pour les définitions de tâche. Pour plus d'informations, consultez the section called "Création d'une définition de tâche à l'aide de la console".	27 octobre 2022
La nouvelle expérience de la console Amazon ECS pour les définitions de tâches est mise à jour	La nouvelle expérience de la console Amazon ECS contient désormais un éditeur JSON pour les définitions de tâche. Pour plus d'informations, consultez the section called "Création d'une définition de tâche à l'aide de la console".	27 octobre 2022
La nouvelle expérience de la console Amazon ECS est mise à jour	La nouvelle expérience de la console Amazon ECS a été mise à jour avec des paramètres de service et de tâche supplémentaires. Pour plus d'informations, consultez the section called "Création d'un déploiement de mise à jour propagée" et the section called "Exécution d'une application en tant que tâche".	7 octobre 2022
Nouvelles informations dans le point de terminaison des métadonnées de tâche version 4	La version 4 du point de terminaison des métadonnées de tâche inclut désormais l'identifiant VPC et le nom du service. Pour plus d'informations, consultez the section called "Point de terminaison des métadonnées de tâches version 4".	7 octobre 2022
Tailles de la nouvelle définition de tâche	Amazon ECS sur Fargate prend désormais en charge les tailles de tâches de 8 vCPU et de 16 vCPU virtuels. Pour plus d'informations, consultez the section called "Taille de la tâche".	16 septembre 2022
Pages CLI ECS archivées	La documentation CLI ECS a été archivée. Nous vous recommandons d'utiliser AWS Copilote pour répondre à vos besoins en matière d'outils de ligne de commande. Pour plus d'informations, consultez Création de ressources Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot.	15 septembre 2022

Modifier	Description	Date
Nouveaux quotas Fargate	Fargate est en train de passer des quotas basés sur le nombre de tâches à vCPU-based quotas. Pour plus d'informations, consultez la section Points de terminaison et quotas Amazon ECS.	8 septembre 2022
Prise en charge des groupes d'instances pré-initialisées pour Amazon EC2 Auto Scaling.	Vous pouvez désormais utiliser les groupes d'instances pré-initialisées Amazon EC2 Auto Scaling pour monter vos applications en puissance plus rapidement et réduire les coûts. Pour plus d'informations, consultez Configuration d'instances préinitialisées pour votre groupe Auto Scaling Amazon ECS.	23 mars 2022
Prise en charge des instances Windows dans ECS Anywhere.	ECS Anywhere prend désormais en charge les instances Windows. Pour plus d'informations, consultez Clusters Amazon ECS pour instances externes.	3 mars 2022
Ajout de la prise en charge ECS Exec pour les instances externes	ECS Exec est désormais pris en charge pour les instances externes. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS avec ECS Exec.	24 janvier 2022
Mise à jour de la nouvelle expérience de la console Amazon ECS	La nouvelle expérience de console Amazon ECS prend en charge la création et la suppression d'un cluster, la mise à jour d'une définition de tâche et l'annulation de l'enregistrement d'une définition de tâche. Pour plus d'informations, consultez Création d'un cluster Amazon ECS pour les charges de travail Fargate, Suppression d'un cluster Amazon ECS., Mise à jour d'une définition de tâche Amazon ECS à l'aide de la console et Annulation de l'enregistrement d'une révision de définition de tâche Amazon ECS à l'aide de la console.	8 décembre 2021

Modifier	Description	Date
Mise à jour de la nouvelle expérience de la console Amazon ECS	La nouvelle expérience de console Amazon ECS prend en charge la création d'une définition de tâche. Pour plus d'informations, consultez Création d'une définition de tâche Amazon ECS à l'aide de la console.	23 novembre 2021
Amazon ECS prend en charge l'architecture ARM 64 bits pour Linux.	Amazon ECS prend en charge l'architecture du processeur ARM 64 bits pour le système d'exploitation Linux. Pour plus d'informations, consultez the section called "Définitions de tâche pour les charges de travail ARM 64 bits".	23 novembre 2021
Prise en charge d'Amazon ECS pour l'option fluentd log-driver-buffer-limit	Amazon ECS prend en charge l'option `log-driver-buffer-limit` fluentd. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	22 novembre 2021
Amazon ECS-optimized Script de génération d'AMI pour Linux	Amazon ECS a ouvert les scripts de génération utilisés pour créer les variantes Linux d'Amazon ECS-optimized AMI. Pour plus d'informations, consultez Amazon ECS-optimized Script de génération d'AMI pour Linux.	19 novembre 2021
État de l'instance de conteneur	Amazon ECS ajoute la prise en charge la surveillance de l'état de l'instances de conteneur. Pour plus d'informations, consultez Surveillance de l'état des instances de conteneurs Amazon ECS.	10 novembre 2021
Prise en charge pour Windows Amazon ECS Exec	Amazon ECS Exec prend en charge Windows. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS avec ECS Exec.	1er novembre 2021
Prise en charge des conteneurs Windows sur Fargate.	Amazon ECS prend en charge les conteneurs Windows sur Fargate. Pour plus d'informations, consultez Journal des modifications de la version de plateforme Windows Fargate.	28 octobre 2021

Modifier	Description	Date
Prise en charge du GPU pour les instances externes sur Amazon ECS Anywhere	Amazon ECS prend en charge la spécification des exigences du GPU dans la définition de tâche pour les tâches exécutées sur des instances externes. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU et Enregistrement d'une instance externe dans un cluster Amazon ECS.	8 octobre 2021
Prise en charge du mode réseau `awsvpc` sous Windows	Amazon ECS prend en charge `awsvpc` sur Windows. Pour plus d'informations, consultez Allocation d'une interface réseau à une tâche Amazon ECS.	15 juillet 2021
Disponibilité générale de Bottlerocket	Amazon ECS prend en charge un Amazon ECS-optimized La variante AMI du système d'exploitation Bottlerocket est disponible en tant qu'AMI. Pour plus d'informations, consultez Amazon ECS-optimized Fusée à bouteilles AMI.	30 Juin 2021
Mise à jour des tâches planifiées d'Amazon ECS	Amazon EventBridge ajout de la prise en charge de paramètres supplémentaires lors de la création de règles qui déclenchent des tâches planifiées Amazon ECS.	25 juin 2021
AWS politiques gérées pour Amazon ECS	Amazon ECS a ajouté de la documentation sur AWS politiques gérées pour les rôles liés aux services. Pour plus d'informations, consultez AWS politiques gérées pour Amazon Elastic Container Service.	8 juin 2021
Commencer avec le AWS CDK	Ajout d'un guide de démarrage pour l'utilisation du AWS CDK avec Amazon ECS. Pour plus d'informa tions, consultez Création de ressources Amazon ECS à l'aide du AWS CDK.	27 mai 2021

Modifier	Description	Date
Amazon ECS n'importe où	Amazon ECS a ajouté la prise en charge de l'enregis trement d'un serveur sur site ou d'une machine virtuelle (VM) auprès de votre cluster. Pour plus d'informations, consultez Clusters Amazon ECS pour instances externes.	25 mai 2021
Amazon ECS-optimized AMI principale Windows Server 20H2	Amazon ECS a ajouté la prise en charge d'un nouvel Amazon Windows ECS-optimized Variante AMI pour Windows Server 20H2 Core. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	19 avril 2021
Amazon ECS Exec	Amazon ECS a publié un nouvel outil de débogage appelé ECS Exec. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS avec ECS Exec.	15 mars 2021
Prise en charge de stratégie de point de terminaison d'un VPC	Amazon ECS prend désormais en charge les stratégie s de point de terminaison Amazon VPC Pour plus d'informations, consultez Création d'une stratégie de point de terminaison d'un VPC pour Amazon ECS.	11 janvier 2021
Nouvelle expérience de console	Amazon ECS a publié une nouvelle expérience de console qui prend en charge la création ou la mise à jour d'un service ou l'exécution d'une tâche autonome. Pour plus d'informations, consultez Création d'un déploiement de mise à jour propagée Amazon ECS et Exécution d'une application en tant que tâche Amazon ECS.	28 décembre 2020
Stratégie de fournisseur de capacité	Amazon ECS prend en charge la mise à jour d'un fournisseur de capacité de groupe Auto Scaling existant.	23 novembre 2020

Modifier	Description	Date
ECS prend désormais en charge les tâches Amazon FSx for Windows File Server pour Windows.	Amazon ECS a ajouté la prise en charge de la spécification de volumes Amazon FSx for Windows File Server dans les définitions de tâches Windows. Pour plus d'informations, consultez Utilisation de FSx for Windows File Server avec Amazon ECS.	11 novembre 2020
Ajout de la prise en charge du mode double pile VPC	Amazon ECS a ajouté la prise en charge de l'utilisation d'un VPC en mode double pile avec des tâches utilisant le mode réseau `awsvpc`, qui prend en charge les adresses IPv6. Pour plus d'informations, consultez Utilisation d'un VPC en mode double pile.	5 novembre 2020
Mise à jour du point de terminaison des métadonnées de tâches v4	Amazon ECS a ajouté des métadonnées supplémentaires à la sortie du point de terminaison v4 des métadonnées de tâche. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4.	5 novembre 2020
Support des Local Zones et des zones de Wavelength	Amazon ECS a ajouté la prise en charge des charges de travail dans les Local Zones et les zones Wavelength. Pour plus d'informations, consultez Applications Amazon ECS dans des sous-réseaux partagés, des zones locales et des Zones Wavelength.	4 septembre 2020
Variante Amazon ECS de l'AMI Bottlerocket	Bottlerocket est un Linux-based système d'exploitation open source spécialement conçu par AWS pour faire fonctionner des conteneurs. Une Amazon ECS-optimized La variante AMI du système d'exploitation Bottlerocket est disponible en tant qu'AMI que vous pouvez utiliser lors du lancement d'instances de conteneur Amazon ECS. Pour plus d'informations, consultez Amazon ECS-optimized Fusée à bouteilles AMI.	31 août 2020

Modifier	Description	Date
Point de terminaison des métadonnées de tâches version 4 mis à jour pour les statistiques de fréquence réseau	Le point de terminaison de métadonnées de tâche version 4 a été mis à jour pour fournir des statistiq ues de débit réseau pour les tâches Amazon ECS qui utilisent les modes réseau `awsvpc` ou `bridge` hébergés sur des instances Amazon EC2 exécutant au moins la version `1.43.0` de l'agent de conteneur. Pour plus d'informations, consultez Point de terminais on des métadonnées de tâches Amazon ECS version 4.	10 août 2020
Métriques d'utilisation de Fargate	AWS Fargate fournit CloudWatch des statistiques d'utilisation qui fournissent une visibilité sur l'utilisa tion de Fargate par vos comptes On-Demand et des ressources sur Fargate Spot. Pour plus d'informations, consultez Métriques d'utilisation.	3 août 2020
AWS Copilot version 0.1.0	Le nouveau AWS Lancement de la CLI Copilot, fournissant des commandes de haut niveau pour simplifier la modélisation, la création, la publication et la gestion d'applications conteneurisées sur Amazon ECS à partir d'un environnement de développement local. Pour plus d'informations, consultez Création de ressources Amazon ECS à l'aide du AWS Interface de ligne de commande Copilot.	9 juillet 2020
AWS Fargate calendrie r de dépréciation des versions de la plateforme	Le calendrier de l'obsolescence des versions de la plateforme Fargate a été ajouté. Pour plus d'informa tions, consultez AWS Dépréciation de la version de la plateforme Fargate Linux.	8 juillet 2020
AWS Expansion de la région de Fargate	Amazon ECS sur AWS Fargate s'est étendue à la région Europe (Milan).	25 juin 2020

Modifier	Description	Date
Publication de l'AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS	Amazon ECS a publié une AMI Amazon Linux 2 (Neuron) optimisée pour Amazon ECS pour les charges de travail inférentes. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	24 juin 2020
Ajout de la prise en charge de la suppressi on des fournisseurs de capacité	Amazon ECS a ajouté la prise en charge de la suppression des fournisseurs de capacité de groupe Auto Scaling.	11 juin 2020
AWS Mise à jour de la version 1.4.0 de la plateforme Fargate	À compter du 28 mai 2020, toute nouvelle tâche Fargate lancée à l'aide de la version 1.4.0 de la plate-forme verra son stockage éphémère de 20 Go crypté avec un AES-256 algorithme de chiffrement à l'aide d'un AWS Fargate-clé de chiffrement gérée. Pour plus d'informations, consultez Stockage éphémère des tâches Fargate pour Amazon ECS.	28 mai 2020
Prise en charge du fichier de variable d'environ nement	Ajout de la prise en charge de la spécification des fichiers de variables d'environnement dans une définition de tâche, ce qui vous permet d'ajouter en bloc des variables d'environnement à vos conteneur s. Pour plus d'informations, consultez Transmiss ion d'une variable d'environnement individuelle à un conteneur Amazon ECS.	18 mai 2020
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu à la région Afrique (Le Cap).	11 mai 2020

Modifier	Description	Date
Quota de service mis à jour	Le quota de service suivant a été mis à jour : • Les clusters par compte ont été élevés de `2,000` à `10,000`. Pour plus d'informations, consultez la section Points de terminaison et quotas Amazon ECS.	17 avril 2020

Modifier	Description	Date
AWS Plateforme Fargate version 1.4.0	AWS La version 1.4.0 de la plate-forme Fargate est publiée et contient les fonctionnalités suivantes : • Ajout de la prise en charge de l'utilisation des volumes de système de fichiers Amazon EFS pour le stockage des tâches permanentes. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS. • Le stockage des tâches éphémères est désormais de 20 Go. Pour plus d'informations, consultez Stockage éphémère des tâches Fargate pour Amazon ECS. • Le comportement du trafic réseau entrant et sortant entre les tâches a été mis à jour. À partir de la version 1.4 de la plateforme, toutes les tâches Fargate reçoivent une interface réseau Elastic unique (appelée ENI de tâche), et tout le trafic réseau passe par cette ENI au sein du VPC et vous est accessible via les journaux de vos flux VPC. Pour plus d'informations, consultez la section Options de mise en réseau des tâches Amazon ECS pour le type de lancement Fargate. • Les ENI des tâches incluent désormais la prise en charge des trames jumbo. Les interfaces réseau sont configurées avec une unité de transmission maximale (MTU), qui correspond à la taille de la charge utile la plus élevée possible dans une seule trame. Plus la MTU est importante, plus la charge utile de l'application peut s'intégrer dans une seule trame. Cela réduit les frais généraux par trame et augmente l'efficacité. La prise en charge des trames	8 avril 2020

Modifier	Description	Date
	jumbo limite la surcharge lorsque le chemin réseau entre votre tâche et la destination prend en charge les trames jumbo, telles que tout le trafic qui reste dans votre VPC. • CloudWatch Container Insights inclura des mesures de performance du réseau pour les tâches Fargate. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée. • Ajout de la prise en charge du point de terminaison de métadonnées de tâche v4, qui fournit des informations supplémentaires pour vos tâches Fargate, y compris les statistiques réseau pour la tâche et la zone de disponibilité dans laquelle la tâche s'exécute. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4. • Ajout de la prise en charge du paramètre `SYS_PTRACE` Linux dans les définitions de conteneur. Pour plus d'informations, consultez Paramètres Linux. • L'agent de conteneur Fargate remplace l'agent de conteneur Amazon ECS pour toutes les tâches Fargate. Cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. • L'environnement d'exécution du conteneur utilise maintenant Containerd au lieu de Docker. Cette modification ne devrait pas avoir d'effet sur la façon dont vos tâches s'exécutent. Vous remarquerez que certains messages d'erreur provenant de l'environ	

Modifier	Description	Date
	nement d'exécution du conteneur mentionneront de moins en moins Docker et de en plus des erreurs générales. Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.	
Prise en charge des systèmes de fichiers Amazon EFS pour les volumes de tâches	Les systèmes de fichiers Amazon EFS peuvent être utilisés comme volumes de données pour vos tâches Amazon ECS et Fargate. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.	8 avril 2020
Point de terminaison des métadonnées de tâches Amazon ECS version 4	À partir de la version 1.39.0 de l'agent de conteneur Amazon ECS et de la version 1.4.0 de la plateform e Fargate, une variable d'environnement nommée `ECS_CONTAINER_METADATA_URI_V4` est injectée dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison des métadonné es de tâches version 4, diverses métadonnées de tâches et statistiques Docker sont disponibles pour les tâches. Pour plus d'informations, consultez Point de terminaison des métadonnées de tâches Amazon ECS version 4.	8 avril 2020
Prise en charge de versions spécifiques de secrets de type Secrets Manager à injecter en tant que variables d'environnement	Ajout de la prise en charge de la spécification de données sensibles à l'aide de versions spécifiques de secrets de type Secrets Manager. Pour plus d'informa tions, consultez Transmission de données sensibles vers un conteneur Amazon ECS.	24 février 2020

Modifier	Description	Date
Ajout d'un supplément CodeDeploy options de configuration de déploiement pour blue/green déploiements	Le CodeDeploy le service a ajouté de nouvelles configurations de déploiement linéaire et Canary pour le type de déploiement Amazon ECS. La possibilité de définir des configurations de déploiement personnalisées est également disponible. Pour plus d'informations, consultez CodeDeploy blue/green déploiements pour Amazon ECS.	6 février 2020
Ajout de l'EFSVolumeConfiguration paramètre de définition de tâche	Le paramètre de définition de tâche `efsVolumeConfiguration` est disponible en version préliminaire publique, ce qui facilite l'utilisation des systèmes de fichiers Amazon EFS avec vos tâches Amazon ECS. Pour plus d'informations, consultez Utilisation des volumes Amazon EFS avec Amazon ECS.	17 janvier 2020
Comportement de journalisation de l'agent de conteneur Amazon ECS mis à jour	Les emplacements de journalisation de l'agent de conteneur Amazon ECS et le comportement de rotation ont été mis à jour. Pour plus d'informations, consultez Paramètres de configuration du journal de l'agent de conteneur Amazon ECS.	13 janvier 2020
Fargate Spot	Amazon ECS a ajouté la prise en charge de l'exécution des tâches en utilisant Fargate Spot. Pour plus d'informations, consultez Clusters Amazon ECS pour Fargate.	3 décembre 2019
Cluster Auto Scaling	L'Auto Scaling de cluster Amazon ECS vous permet de mieux contrôler la façon dont vous mettez à l'échelle les tâches au sein d'un cluster. Pour plus d'informations, consultez Gestion automatique de la capacité Amazon ECS grâce à l'autoscaling de cluster.	3 décembre 2019

Modifier	Description	Date
Fournisseurs de capacité de cluster	Les fournisseurs de capacité de cluster Amazon ECS déterminent l'infrastructure à utiliser pour vos tâches. Pour plus d'informations, consultez Clusters Amazon ECS.	3 décembre 2019
Création d'un cluster sur un AWS Outposts	Amazon ECS prend désormais en charge la création de clusters sur un AWS Outposts. Pour plus d'informations, consultez the section called "Amazon Elastic Container Service sur AWS Outposts".	3 décembre 2019
Événements d'action de service	Amazon ECS envoie désormais des événements à Amazon EventBridge lorsque certaines opérations de maintenance se produisent. Pour plus d'informations, consultez Événements d'action d'un service Amazon ECS.	25 novembre 2019
Amazon ECS GPU-optimized AMI prend en charge les instances G4	Amazon ECS a ajouté la prise en charge de la famille de types d'instance g4 lors de l'utilisation d'Amazon ECS GPU-optimized AMI. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.	8 octobre 2019
FireLens pour Amazon ECS	FireLens pour Amazon ECS est en disponibilité générale. FireLens pour Amazon ECS vous permet d'utiliser des paramètres de définition de tâches pour acheminer les journaux vers un AWS destination du service ou du partenaire pour le stockage et l'analyse des journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	30 septembre 2019
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu aux régions Europe (Paris), Europe (Stockholm) et Moyen-Orient (Bahreïn).	30 septembre 2019

Modifier	Description	Date
Deep Learning Containers avec Elastic Inference sur Amazon ECS	Amazon ECS prend en charge l'association d'accélérateurs Amazon Elastic Inference à vos conteneurs afin d'améliorer l'efficacité des charges de travail d'inférence de deep learning en cours d'exécution.	3 septembre 2019
FireLens pour Amazon ECS	FireLens pour Amazon ECS est disponible en version préliminaire publique. FireLens pour Amazon ECS vous permet d'utiliser des paramètres de définition de tâches pour acheminer les journaux vers un AWS destination du service ou du partenaire pour le stockage et l'analyse des journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à un AWS service ou AWS Partner.	30 août 2019
CloudWatch Informations sur les conteneurs	CloudWatch Container Insights est désormais disponible pour tous. Il vous permet de collecter, regrouper et récapituler les métriques et les journaux de vos applications et microservices conteneurisés. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.	30 août 2019
Configuration de l'échange au niveau du conteneur	Amazon ECS prend désormais en charge le contrôle de l'utilisation de l'espace mémoire d'échange sur vos instances de conteneur Linux au niveau du conteneur. En utilisant une configuration d'échange par conteneur, l'échange peut être activé ou désactivé pour chaque conteneur au sein d'une définition de tâche, et pour les conteneurs pour lesquels l'échange est activé, la quantité maximale d'espace d'échange utilisée peut être limitée. Pour plus d'informations, consultez Gestion de l'espace mémoire d'échange de conteneur sur Amazon ECS.	16 août 2019

Modifier	Description	Date
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu à la région Asie-Pacifique (Hong Kong).	6 août 2019
Jonction Elastic Network Interface	Prise en charge de types d'instance Amazon EC2 supplémentaires pour la fonction de jonction d'ENI. Pour plus d'informations, consultez Instances prises en charge pour augmenter le nombre d'interfaces réseau de conteneurs Amazon ECS.	1 août 2019
Enregistrement de plusieurs groupes cibles auprès d'un service	Ajout de la prise en charge de la spécification de plusieurs groupes cibles dans une définition de service. Pour plus d'informations, consultez Enregistrement de plusieurs groupes cibles auprès d'un service Amazon ECS.	30 juillet 2019
Spécification de données sensibles à l'aide de secrets de Secrets Manager	Ajout d'un didacticiel pour la spécification de données sensibles à l'aide de secrets de Secrets Manager. Pour plus d'informations, consultez Spécification de données sensibles à l'aide de secrets Secrets Manager dans Amazon ECS.	20 juillet 2019
CloudWatch Informations sur les conteneurs	Amazon ECS a ajouté la prise en charge de CloudWatch Informations sur les conteneurs. Pour plus d'informations, consultez Surveillance des conteneurs Amazon ECS au moyen de Container Insights avec observabilité améliorée.	9 juillet 2019
Resource-level autorisations pour les services et les ensembles de tâches Amazon ECS	Amazon ECS a étendu la prise en charge des autorisations au niveau des ressources pour les services et ensembles de tâches Amazon ECS. Pour plus d'informations, consultez Fonctionnement d'Amazon Elastic Container Service avec IAM.	27 juin 2019
Nouvel Amazon ECS-optimized AMI patché pour AWS-2005-2019	Amazon ECS a mis à jour l'Amazon ECS-optimized AMI pour corriger les vulnérabilités décrites dans AWS-2005-2019.	17 juin 2019

Modifier	Description	Date
Jonction Elastic Network Interface	Amazon ECS introduit la prise en charge du lancement des instances de conteneur grâce à des types d'instances Amazon EC2 pris en charge qui ont augmenté la densité d'interface réseau Elastic (ENI). L'utilisation de ces types d'instances et du paramètre de compte `awsvpcTrunking` permet d'améliorer la densité ENI sur les instances de conteneur nouvellem ent lancées et de placer plus de tâches sur chaque instance de conteneur. Pour plus d'informations, consultez Augmentation des interfaces réseau d'une instance de conteneur Amazon ECS Linux.	6 juin 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le pilote de journal `splunk` en plus du pilote de journal `awslogs`. Pour plus d'informa tions, consultez Stockage et journalisation.	1er mai 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 1er mai 2019, toute nouvelle tâche Fargate lancée prend en charge le référencement des données sensibles dans la configuration de journal d'un conteneur à l'aide du paramètre de définition de conteneur `secretOptions` . Pour plus d'informa tions, consultez Transmission de données sensibles vers un conteneur Amazon ECS.	1er mai 2019
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	À compter du 2 avril 2019, toute nouvelle tâche Fargate lancée permet d'injecter des données sensibles dans vos conteneurs en stockant vos données sensibles dans AWS Secrets Manager secrets ou AWS Systems Manager Paramètre Stockez les paramètres, puis référencez-les dans la définition de votre conteneur. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.	2 avril 2019

Modifier	Description	Date
AWS Mise à jour de la version 1.3.0 de la plateforme Fargate	Depuis le 27 mars 2019, toute nouvelle tâche Fargate lancée peut utiliser des paramètres de définition de tâches supplémentaires qui vous permettent de définir une configuration proxy, des dépendances de démarrage et d'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de démarrage par conteneur. Pour plus d'informations, consultez Configuration du proxy, Dépendances du conteneur et Temporisations de conteneurs.	27 mars 2019
Amazon ECS introduit le type de déploiement externe	Le type de déploiement externe vous permet d'utilise r n'importe quel contrôleur de déploiement tiers pour un contrôle complet du processus de déploiement pour un service Amazon ECS service. Pour plus d'informa tions, consultez Déploiement de services Amazon ECS à l'aide d'un contrôleur tiers.	27 mars 2019
AWS Conteneurs de Deep Learning sur Amazon ECS	AWS Les conteneurs Deep Learning sont un ensemble d'images Docker destinées à la formation et à la diffusion de modèles dans TensorFlow sur Amazon Elastic Container Service (Amazon ECS). Les conteneurs Deep Learning fournissent des environne ments optimisés avec TensorFlow, les bibliothèques Nvidia CUDA (pour les instances GPU) et Intel MKL (pour les instances CPU) et sont disponibles sur Amazon ECR. Pour plus d'informations, consultez Utilisation AWS Deep Learning Containers sur Amazon ECS.	27 mars 2019

Modifier	Description	Date
Amazon ECS présente une gestion améliorée des dépendances de conteneurs	Amazon ECS présente des paramètres supplémentaires de définition de tâche qui vous permettent de définir des dépendances pour le démarrage et l'arrêt de conteneurs, ainsi qu'une valeur de temporisation d'arrêt et de lancement pour chaque conteneur. Pour plus d'informations, consultez Dépendances du conteneur.	7 mars 2019
Amazon ECS présente l'API `PutAccountSettingDefault`	Amazon ECS présente le `PutAccountSettingDefault` API qui permet à un utilisateur de définir la valeur par défaut ARN/ID statut d'activation du format pour tous les utilisateurs et rôles du compte. Auparavant, le statut d'acceptation par défaut du compte nécessitait d'utiliser le propriétaire du compte. Pour plus d'informations, consultez Amazon Resource Names (ARN) et ID.	8 février 2019
Amazon ECS prend en charge les charges de travail GPU	Amazon ECS introduit la prise en charge des charges de travail GPU en vous permettant de créer des clusters avec GPU-enabled instances de conteneurs. Dans une définition de tâche, vous pouvez spécifier le nombre de GPU requis et l'agent ECS épinglera les GPU physiques au conteneur. Pour plus d'informations, consultez Définitions de tâches Amazon ECS pour les charges de travail GPU.	4 février 2019
Prise en charge étendue des secrets par Amazon ECS	Support étendu d'Amazon ECS pour l'utilisation AWS Secrets Manager secrets directement dans vos définitions de tâches pour injecter des données sensibles dans vos conteneurs. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.	21 janvier 2019

Modifier	Description	Date
Points de terminaison d'un VPC d'interface (AWS PrivateLink)	Ajout de la prise en charge pour la configuration de points de terminaison VPC optimisés par AWS PrivateLink. Cela vous permet de créer une connexion privée entre votre VPC et Amazon ECS sans avoir besoin d'un accès Internet, via une instance NAT, une connexion VPN ou Direct Connect. Pour plus d'informations, veuillez consulter Points de terminaison d'un VPC d'interface (AWS PrivateLink).	26 décembre 2018
AWS Plateforme Fargate version 1.3.0	Nouveau AWS Version de la plateforme Fargate publiée, qui contient : • Ajout d'un support pour l'utilisation AWS Systems Manager Paramètre Stockez les paramètres pour injecter des données sensibles dans vos conteneurs. Pour plus d’informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS. • Ajout du recyclage de tâche pour les tâches Fargate, qui est le processus d'actualisation des tâches qui font partie d'un service Amazon ECS service. Pour plus d'informations, voir Retrait et maintenance des tâches pour AWS Fargate sur Amazon ECS. Pour plus d’informations, consultez Versions de plateforme Fargate pour Amazon ECS.	le 17 décembre 2018

Modifier	Description	Date
Mise à jour de Service Limits	Les limites de service suivantes ont été mises à jour : • Le nombre de clusters par région et par compte a été augmenté de `1000` à `2000`. • Le nombre d'instances de conteneur par cluster a été augmenté de `1000` à `2000`. • Le nombre de services par cluster a été augmenté de `500` à `1000`. Pour plus d'informations, consultez Quotas de service Amazon ECS service.	14 décembre 2018
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu aux régions Asie-Pacifique (Mumbai) et Canada (Centre). Pour plus d'informations, consultez Régions prises en charge pour Amazon ECS sur AWS Fargate.	7 décembre 2018
Amazon ECS blue/green déploiements	Amazon ECS a ajouté le support pour blue/gree n déploiements utilisant CodeDeploy. Ce type de déploiement vous permet de vérifier un nouveau déploiement d'un service avant de lui envoyer un trafic de production. Pour plus d'informations, consultez CodeDeploy blue/ green déploiements pour Amazon ECS.	27 novembre 2018
Amazon ECS-optimized Sortie de l'AMI Amazon Linux 2 (arm64)	Amazon ECS a publié un Amazon ECS-optimized AMI Amazon Linux 2 pour architecture arm64. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	26 novembre 2018

Modifier	Description	Date
Ajout de la prise en charge d'indicateurs Docker supplémentaires dans les définitions de tâche.	Amazon ECS a introduit la prise en charge des indicateurs Docker suivants dans les définitions de tâche : • Mode IPC • Mode PID	16 novembre 2018
Prise en charge d'Amazon ECS	Amazon ECS a ajouté la prise en charge de l'utilisation AWS Systems Manager Paramètre Stockez les paramètres pour injecter des données sensibles dans vos conteneurs. Pour plus d'informations, consultez Transmission de données sensibles vers un conteneur Amazon ECS.	15 novembre 2018
Identification des ressources	Amazon ECS comprend la prise en charge de l'ajout de balises de métadonnées à vos services, définitions de tâche, tâches, clusters et instances de conteneur. Pour plus d'informations, consultez Balisage des ressources Amazon ECS.	15 novembre 2018
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu aux régions de l'Ouest des États-Unis (Californie du Nord) et de l'Asie-Pacifique (Séoul). Pour plus d'informations, consultez Architecte pour AWS Fargate pour Amazon ECS.	7 novembre 2018

Modifier	Description	Date
Mise à jour de Service Limits	Les limites de service suivantes ont été mises à jour : • Le nombre de tâches utilisant le type de lancement Fargate par région et par compte est passé de `20` à `50`. • Le nombre d'adresses IP publiques pour les tâches utilisant le type de lancement Fargate est passé de `20` à `50`. Pour plus d'informations, consultez Quotas de service Amazon ECS service.	31 octobre 2018
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu à la région Europe (Londres). Pour plus d'informations, consultez Architecte pour AWS Fargate pour Amazon ECS.	26 octobre 2018
Amazon ECS-optimized Sortie de l'AMI Amazon Linux 2	Amazon ECS propose deux versions des AMI Linux optimisées pour le service. La version la plus récente et recommandée est basée sur x;. Amazon ECS propose également des AMI basées sur , mais nous vous recommandons de migrer vos charges de travail vers la variante Amazon Linux 2, car la prise en charge de l'AMI Linux Amazon prendra fin le 30 juin 2020. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	18 octobre 2018

Modifier	Description	Date
Point de terminaison des métadonnées de tâches Amazon ECS version 3	À partir de la version 1.21.0 de l'agent de conteneur Amazon ECS, l'agent injecte une variable d'environnement appelée `ECS_CONTAINER_METADATA_URI` dans chaque conteneur d'une tâche. Lorsque vous interrogez le point de terminaison de la version 3 des métadonnées de tâche, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Surveillance des applications à l'aide des métadonnées Amazon ECS.	18 octobre 2018
Expansion de la région de découverte de service Amazon ECS service	La prise en charge de la découverte de service Amazon ECS service a été étendue aux régions Canada (Centre), Amérique du Sud (São Paulo), Asie-Pacifique (Séoul), Asie-Pacifique (Mumbai) et Europe (Paris). Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.	le 27 septembre 2018
Ajout de la prise en charge d'indicateurs Docker supplémentaires dans les définitions de conteneur.	Amazon ECS a introduit la prise en charge des indicateurs Docker suivants dans les définitions de conteneur : • Contrôles système • Interactive • Pseudo Terminal	17 septembre 2018

Modifier	Description	Date
Prise en charge de l'authentification de registre privé pour Amazon ECS à l'aide de AWS Tâches Fargate	Amazon ECS a introduit la prise en charge des tâches Fargate utilisant l'authentification de registre privé à l'aide d' AWS Secrets Manager. Cette fonction vous permet de stocker vos informations d'identification en toute sécurité, puis de les référencer dans votre définition de conteneur, ce qui permet à vos tâches d'utiliser des images privées. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.	10 septembre 2018
Expansion de la région de découverte de service Amazon ECS service	La prise en charge de la découverte de service Amazon ECS service a été étendue aux régions Asie-Pacifique (Singapour), Asie-Pacifique (Sydney), Asie-Pacifique (Tokyo), UE (Francfort) et Europe (Londres). Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.	30 août 2018
Prise en charge des tâches planifiées pour les tâches Fargate	Amazon ECS a introduit la prise en charge des tâches planifiées pour le type de lancement Fargate.	28 août 2018
Authentification de registre privé à l'aide AWS Secrets Manager soutenir	Amazon ECS a introduit la prise en charge de l'authentification de registre privé à l'aide d' AWS Secrets Manager. Cette fonction vous permet de stocker vos informations d'identification en toute sécurité, puis de les référencer dans votre définition de conteneur, ce qui permet à vos tâches d'utiliser des images privées. Pour plus d'informations, consultez En utilisant des produits non AWS images de conteneurs dans Amazon ECS.	16 août 2018

Modifier	Description	Date
Ajout de la prise en charge des volumes Docker	Amazon ECS a introduit la prise en charge des volumes Docker. Pour plus d'informations, consultez Options de stockage pour les tâches Amazon ECS.	9 août 2018
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu aux régions Europe (Francfort), Asie-Pacifique (Singapour) et Asie-Pacifique (Sydney). Pour plus d'informations, consultez Architecte pour AWS Fargate pour Amazon ECS.	19 juillet 2018

Modifier	Description	Date
Ajout des stratégies de planificateur de service Amazon ECS service	Amazon ECS a introduit le concept de stratégies de planificateur de service. Deux stratégies de planificateur de service sont disponibles : • `REPLICA` – La stratégie de planification des réplicas place et gère le nombre de tâches souhaité dans votre cluster. Par défaut, le planificateur de service répartit les tâches entre les zones de disponibilité. Vous pouvez utiliser des stratégies et des contraint es de placement des tâches afin de personnaliser la façon dont les tâches sont placées. Pour plus d'informations, consultez Stratégie de planification de réplica. • `DAEMON` – La stratégie de planification du démon déploie exactement une tâche sur chaque instance de conteneur active qui répond à toutes les contraint es de placement des tâches spécifiées dans votre cluster. Lors de l'utilisation de cette stratégie, il n'est pas nécessaire de spécifier un nombre de tâches souhaité, une stratégie de placement des tâches ou d'utiliser les politiques Service Auto Scaling. Pour plus d'informations, consultez Stratégie de planification de démon. **Note** Les tâches Fargate ne prennent pas en charge la stratégie de planification `DAEMON`. .	12 juin 2018

Modifier	Description	Date
Agent de conteneur Amazon ECS v1.18.0	Nouvelle version de l'agent de conteneur Amazon ECS, qui intègre les nouvelles fonctionnalités suivantes : • Ajout de la prise en charge de la personnalisation du comportement d'extraction d'image de l'agent de conteneur à l'aide du paramètre `ECS_IMAGE_PULL_BEHAVIOR` . Pour plus d'informations, consultez Configuration de l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez amazon-ecs-agent github.	24 mai 2018
Ajout de la prise en charge des modes réseau `bridge` et de `host` lors de la configuration de la découverte de service	Ajout de la prise en charge de la configuration de la découverte de service pour les services Amazon ECS à l'aide de définitions de tâches qui spécifient les modes réseau `bridge` ou `host`. Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.	le 22 mai 2018
Ajout de la prise en charge d'Amazon supplémentaires ECS-optimized Paramètres de métadonnées AMI	Ajout de sous-paramètres qui vous permettent de récupérer par programmation l'Amazon ECS-optimized ID AMI, nom de l'image, système d'exploitation, version de l'agent de conteneur et version d'exécution. Interrogation des métadonnées via l'API Systems Manager Parameter Store. Pour plus d'informations, consultez Récupération d'Amazon ECS-optimized Métadonnées AMI Linux.	le 9 mai 2018

Modifier	Description	Date
AWS Expansion de la région de Fargate	AWS Fargate avec Amazon ECS s'est étendu aux régions de l'est des États-Unis (Ohio), de l'ouest des États-Unis (Oregon) et de l'ouest de l'UE (Irlande). Pour plus d'informations, consultez Architecte pour AWS Fargate pour Amazon ECS.	26 avril 2018
Amazon ECS-optimized Récupération des métadonnées AMI	Ajout de la possibilité de récupérer Amazon par programmation ECS-optimized Métadonnées AMI à l'aide de l'API Systems Manager Parameter Store. Pour plus d'informations, consultez Récupération d'Amazon ECS-optimized Métadonnées AMI Linux.	10 avril 2018
AWS Version de la plateforme Fargate	Nouveau AWS Version de la plateforme Fargate publiée, qui contient : • Ajout de la prise en charge de Surveillance des applications à l'aide des métadonnées Amazon ECS. • Ajout de la prise en charge de Surveillance de l'état. • Ajout de la prise en charge de Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS Pour plus d'informations, consultez Versions de plateforme Fargate pour Amazon ECS.	26 mars 2018
Découverte d'Amazon ECS service	Ajout de l'intégration à Route 53 pour la prise en charge de la découverte de service Amazon ECS service. Pour plus d'informations, consultez Utilisation de la découverte de service pour connecter les services Amazon ECS avec des noms DNS.	22 mars 2018

Modifier	Description	Date
Prise en charge des paramètres Docker shm-size et tmpfs	Ajout de la prise en charge des paramètres Docker shm-size et tmpfs dans les définitions de tâches Amazon ECS. Pour en savoir plus sur la syntaxe mise à jour de la CLI ECS, veuillez consulter Paramètres Linux.	le 20 mars 2018
Surveillances de l'état du conteneur	Ajout de la prise en charge des surveillances de l'état Docker dans les définitions de conteneur. Pour plus d'informations, consultez Surveillance de l'état.	8 mars 2018
AWS Fargate	Ajout d'une vue d'ensemble pour Amazon ECS avec AWS Fargate. Pour plus d'informations, consultez Architecte pour AWS Fargate pour Amazon ECS.	22 février 2018
Point de terminaison des métadonnées de tâches Amazon ECS	À partir de la version 1.17.0 de l'agent de conteneur Amazon ECS, différentes métadonnées de tâche et statistiques Docker sont disponibles pour les tâches qui utilisent le mode réseau `awsvpc` à un point de terminaison HTTP fourni par l'agent de conteneur Amazon ECS. Pour plus d'informations, consultez Surveillance des applications à l'aide des métadonnées Amazon ECS.	8 février 2018
Auto Scaling d'Amazon ECS service à l'aide des stratégies de suivi de la cible	Ajout de la prise en charge de l'Auto Scaling d'Amazon ECS service à l'aide des stratégies de suivi de la cible dans la console Amazon ECS. Pour plus d'informations, consultez Utilisation d'une métrique cible pour mettre à l'échelle les services Amazon ECS. Suppression du précédent didacticiel portant sur la mise à l'échelle par étapes dans l'assistant de première exécution d'ECS. Ce didacticiel a été remplacé par un nouveau didacticiel dédié au suivi de la cible.	8 février 2018

Modifier	Description	Date
Prise en charge de Docker 17.09	Ajout de la prise en charge de Docker 17.09. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	18 janvier 2018
Nouveau comportement du planificateur de service	Mise à jour des informations sur le comportement des tâches du service dont le lancement a échoué. Nouveau message d'événement documenté qui se déclenche après deux échecs de lancement consécutifs d'une tâche du service.	11 janvier 2018
Période d'attente pour l'initialisation de la surveillance de l'état Elastic Load Balancing	Ajout de la possibilité de spécifier un délai d'attente pour les vérifications de l'état.	27 décembre 2017
Task-level CPU et mémoire	Ajout de la prise en charge de la spécification de l'UC et de la mémoire de niveau de tâche dans les définitions de tâche. Pour plus d'informations, consultez TaskDefinition.	12 décembre 2017
Rôle d'exécution de tâche	L'agent de conteneur Amazon ECS effectue des appels aux actions d'API Amazon ECS à votre place. Il nécessite donc un rôle et une stratégie IAM pour que le service sache que l'agent vous appartient. Les actions suivantes sont couvertes par le rôle d'exécution des tâches : • Appels à Amazon ECR pour extraire l'image de conteneur • Appels à CloudWatch pour stocker les journaux des applications de conteneurs Pour plus d'informations, consultez Rôle IAM d'exécution de tâche Amazon ECS.	7 décembre 2017

Modifier	Description	Date
Prise en charge des conteneurs Windows par GA	Ajout de la prise en charge des conteneurs Windows 2016. Pour plus d'informations, consultez Amazon ECS-optimized Variantes AMI.	5 décembre 2017
AWS Fargate, Géorgie	Ajout de la prise en charge du lancement des services Amazon ECS avec le type de lancement Fargate. Pour plus d'informations, consultez Architecture de votre solution pour Amazon ECS.	29 novembre 2017
Changement de nom pour Amazon ECS	Nouveau nom pour Amazon Elastic Container Service (anciennement Amazon EC2 Container Service).	21 novembre 2017
Mise en réseau des tâches	Les fonctions de mise en réseau des tâches fournies par le mode réseau attribuent aux tâches Amazon ECS les mêmes propriétés de mise en réseau que les instances Amazon EC2. Lorsque vous utilisez le mode réseau awsvpc dans vos définitions de tâches, chaque tâche qui est lancée à partir de cette définition de tâche obtient sa propre interface réseau Elastic, une adresse IP privée principale ainsi qu'un nom d'hôte DNS interne. La fonctionnalité de mise en réseau des tâches simplifie la mise en réseau des conteneurs et vous procure plus de contrôle sur la façon dont les applications conteneurisées communiquent entre elles et avec les autres services dans vos VPC. Pour plus d'informations, consultez Options de mise en réseau des tâches Amazon ECS pour EC2.	14 novembre 2017
Fichier de métadonnées de conteneur Amazon ECS	Les conteneurs Amazon ECS peuvent désormais accéder aux métadonnées, comme leur conteneur Docker ou ID d'image, la configuration de mise en réseau ou les ARN Amazon. Pour plus d'informations, consultez Fichier de métadonnées de conteneur Amazon ECS.	2 novembre 2017

Modifier	Description	Date
Prise en charge de Docker 17.06	Ajout de la prise en charge de Docker 17.06. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	2 novembre 2017
Prise en charge des indicateurs Docker : device et init	Ajout de la prise en charge des indicateurs device et init de Docker dans les définitions de tâches utilisant le paramètre `LinuxParameters` (`devices` et `initProcessEnabled`). Pour plus d'informations, consultez LinuxParameters.	2 novembre 2017
Prise en charge des indicateurs Docker : cap-add et cap-drop	Ajout de la prise en charge des indicateurs cap-add et cap-drop Docker dans les définitions de tâches utilisant le paramètre `LinuxParameters` (`capabilities`). Pour plus d'informations, consultez LinuxParameters.	22 septembre 2017
Prise en charge de Network Load Balancer	Amazon ECS a ajouté la prise en charge des équilibreurs de charge Network Load Balancers dans la console Amazon ECS.	7 septembre 2017
RunTask remplacements	Ajout de la prise en charge des remplacements de définitions de tâches lors de l'exécution d'une tâche. Cela vous permet d'exécuter une tâche tout en modifiant une définition de tâche, sans avoir à configurer une nouvelle révision de définition de tâche. Pour plus d'informations, consultez Exécution d'une application en tant que tâche Amazon ECS.	27 juin 2017
Mise à jour des tâches planifiées d'Amazon ECS	Ajout de la prise en charge de la planification de tâches avec cron.	7 juin 2017
Instances Spot dans la console Amazon ECS	Ajout de la prise en charge de la création d'instances de conteneur de parc d'instances Spot dans la console Amazon ECS. Pour plus d'informations, consultez Lancement d'une instance de conteneur Amazon ECS Linux.	6 juin 2017

Modifier	Description	Date
Notification Amazon SNS pour le nouvel Amazon ECS-optimized Communiqués AMI	Ajout de la possibilité de s'abonner aux notifications SNS concernant le nouvel Amazon ECS-optimized Communiqués AMI.	23 mars 2017
Microservices et tâches par lots	Ajout de la documentation sur deux cas d'utilisation courants d'Amazon ECS : les microservices et les tâches par lots.	Février 2017
Drainage des instances de conteneur	Ajout de la prise en charge du drainage des instances de conteneur, une méthode permettant de supprimer des instances de conteneur d'un cluster. Pour plus d'informations, consultez Drainage des instances de conteneur Amazon ECS.	24 janvier 2017
Prise en charge de Docker 1.12	Ajout de la prise en charge de Docker 1.12. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	24 janvier 2017
Nouvelles stratégies de placement de tâches	Ajout de la prise en charge de stratégies de placement de tâches : placement en fonction de l'attribut, BinPack, répartition par zone de disponibilité et une tâche par hôte. Pour plus d'informations, consultez Utilisation des stratégies pour définir le placement des tâches Amazon ECS.	29 décembre 2016
Prise en charge des conteneurs Windows en version bêta	Ajout de la prise en charge des conteneurs Windows Server 2016 (version bêta). Pour plus d'informations, consultez Amazon ECS-optimized Variantes AMI.	20 décembre 2016
Prise en charge de Blox OSS	Ajout de la prise en charge de Blox OSS, qui permet de planifier des tâches personnalisées. Pour plus d'informations, consultez Planification de vos conteneurs sur Amazon ECS.	1 décembre 2016

Modifier	Description	Date
Diffusion de l'événement Amazon ECS pour CloudWatch Évènements	Amazon ECS envoie désormais les modifications d'état des instances de conteneurs et des tâches à CloudWatch Évènements. Pour plus d'informations, consultez Automatisez les réponses aux erreurs d'Amazon ECS à l'aide de EventBridge.	21 novembre 2016
Conteneur Amazon ECS connecté à CloudWatch Journaux	Ajout de la prise en charge du pilote awslogs pour envoyer des flux de journaux de conteneurs à CloudWatch Journaux. Pour plus d'informations, consultez Envoyer les journaux Amazon ECS à CloudWatch.	12 septembre 2016
Services Amazon ECS avec prise en charge de l'Elastic Load Balancing pour les ports dynamiques	Ajout d'un équilibreur de charge pour prendre en charge plusieurs combinaisons instance:port par écouteur, ce qui augmente la flexibilité des conteneurs. Désormais, vous pouvez autoriser Docker à définir dynamiquement le port hôte du conteneur et le planificateur ECS enregistre la combinaison instance:port avec l'équilibreur de charge. Pour plus d'informations, consultez Utilisation de l'équilibrage de charge pour répartir le trafic des services Amazon ECS.	11 août 2016
Rôles IAM pour les tâches Amazon ECS	Ajout de la prise en charge de l'association des rôles IAM avec une tâche. Cela fournit aux conteneurs des autorisations plus détaillées qu'un rôle unique pour une instance de conteneur complète. Pour plus d'informations, consultez rôle IAM de tâche Amazon ECS.	13 juillet 2016
Prise en charge de Docker 1.11	Ajout de la prise en charge de Docker 1.11. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	31 mai 2016
Mise à l'échelle automatique des tâches	Amazon ECS a ajouté la prise en charge de la mise à l'échelle automatique de vos tâches exécutées par un service. Pour plus d'informations, consultez Mise à l'échelle automatique de votre service Amazon ECS.	18 mai 2016

Modifier	Description	Date
Filtrage des définitions de tâches par famille de tâches	Ajout de la prise en charge du filtrage d'une liste de définitions de tâches par famille de définitions de tâches. Pour plus d'informations, consultez ListTaskDefinitions.	17 mai 2016
Journalisation des conteneurs Docker et des agents Amazon ECS	Amazon ECS a ajouté la possibilité d'envoyer les journaux de l'agent ECS et du conteneur Docker depuis des instances de conteneur vers CloudWatch Journaux pour simplifier la résolution des problèmes.	5 mai 2016
ECS-optimized AMI prend désormais en charge Amazon Linux 2016.03.	Le ECS-optimized AMI a ajouté la prise en charge d'Amazon Linux 2016.03. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	5 avril 2016
Prise en charge de Docker 1.9	Ajout de la prise en charge de Docker 1.9. Pour plus d'informations, consultez Amazon ECS-optimized AMI Linux.	22 décembre 2015
CloudWatch mesures pour la réservation du processeur et de la mémoire du cluster	Amazon ECS a ajouté une version personnalisée CloudWatch métriques pour la réservation du processeur et de la mémoire.	22 décembre 2015
Nouvelle expérience de mise en route Amazon ECS	Ajout de la création de rôles zéro clic lors de la première exécution de la console Amazon ECS.	23 novembre 2015
Placement des tâches dans les zones de disponibilité	Ajout de la prise en charge du placement des tâches dans les zones de disponibilité par le planificateur de services Amazon ECS service.	8 octobre 2015

Modifier	Description	Date
CloudWatch mesures pour les clusters et services Amazon ECS	Amazon ECS a ajouté une version personnalisée CloudWatch des mesures relatives à l'utilisation du processeur et de la mémoire pour chaque instance de conteneur, service et famille de définitions de tâches d'un cluster. Ces nouvelles mesures peuvent être utilisées pour redimensionner les instances de conteneurs d'un cluster à l'aide de groupes Auto Scaling ou pour créer des groupes personnalisés CloudWatch alarmes.	17 août 2015
Prise en charge des ports UDP	Ajout de la prise en charge des ports UDP dans les définitions de tâches.	7 juillet 2015
Remplacements de variable d'environnement	Ajout du support pour le désenregistrementTaskDefinition et des remplacements de variables d'environnement pour RunTask.	18 juin 2015
Automatisation des mises à jour de l'agent Amazon ECS	Ajout de la possibilité de voir la version de l'agent ECS en cours d'exécution sur une instance de conteneur. Possibilité également de mettre à jour l'agent ECS à partir du Console de gestion AWS, AWS CLI et le SDK.	11 juin 2015
Planificateur de service Amazon ECS service et intégration d'Elastic Load Balancing	Ajout de la possibilité de définir un service et de l'associer à un équilibreur de charge Elastic Load Balancing.	9 avril 2015
Disponibilité générale d'Amazon ECS	Disponibilité générale d'Amazon ECS dans les régions USA Est (Virginie du Nord), USA Ouest (Oregon), Asie-Pacifique (Tokyo) et UE (Irlande).	9 avril 2015

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.